Master's thesis
Theoretical and Computational Methods

GPU implementation of wet foam model and the origin of phase separation

Juhana Lankinen

March 15, 2020

Supervisor(s): Dr. Antti Puisto

Examiner(s): Prof. Kai Nordlund
Prof. Mikko Alava

University of Helsinki
Faculty of Science

PL 64 (Gustaf Hällströmin katu 2a)
00014 Helsingin yliopisto

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Theoretical and Computational Methods

Tekijä — Författare — Author

Juhana Lankinen

Työn nimi — Arbetets titel — Title

GPU implementation of wet foam model and the origin of phase separation

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	March 15, 2020	67

Tiivistelmä — Referat — Abstract

Due to the unique properties of foams, they can be found in many different applications in a wide variety of fields. The study of foams is also useful for the many properties they share with other phenomena, like impurities in cooling metals, where the impurities coarsen similarly to bubbles in foams. For these and other reasons foams have been studied extensively for over a hundred years and continue being an interesting area of study today due to new insights in both experimental and theoretical work and new applications waiting to be used and realized in different industries. The most impactful early work in the study of the properties of foams was done in the late 1800s by Plateau. His work was extended in the early to mid-1900s by Lifshitz, Slyozov, Wagner and von Neumann and by many more authors in recent years. The early work was mostly experimental or theoretical in the sense of performing mathematical calculations on paper, while the modern methods of study have kept the experimental part – with more refined methods of measurement of course – but shifted towards the implementation of the theory as simulations instead of solving problems on paper. In the early 90s Durian proposed a new method for simulating the mechanics of wet foams, based on repulsive spring-like forces between neighboring bubbles. This model was later extended to allow for the coarsening of the foam, and a slightly changed version of this model has been implemented in the code presented in this thesis. As foams consist of a very large number of bubbles, it is important to be able to simulate sufficiently large systems to realistically study the physics of foams. Very large systems have traditionally been too slow to simulate on the individual bubble level in the past, but thanks to the popularity of computer games and the continuous demand for better graphics in games, the graphics processing units have become very powerful and can nowadays be used to do highly parallel general computing. In this thesis, a modified version of Durian's wet foam model that runs on the GPU is presented. The code has been implemented in modern C++ using Nvidia's CUDA on the GPU. Using this program first a typical two-dimensional foam is simulated with 10^5 bubbles. It is found that the simulation code replicates the expected behaviour for this kind of foam. After this, a more detailed analysis is done of a novel phenomenon of the separation of liquid and gas phases in low gas fraction foams that arises only with sufficiently large system sizes. It is found that the phase separation causes the foam to evolve as would a foam of higher gas fraction until the phases have mixed back together. It is hypothesized that the reason causing the phase separation is related to uneven energy distribution in the foam, which itself is related to jamming and uneven distribution of the sizes of the bubbles in the foam.

Avainsanat — Nyckelord — Keywords

GPU, wet foam, coarsening, phase separation

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Here, for instance, we have a book, in two volumes, octavo, written by a distinguished man of science, and occupied for the most part with the theory and practice of bubble-blowing. Can the poetry of bubbles survive this?

— James Clerk Maxwell on Joseph Plateau's 1873 publication.

Contents

1. Introduction

Bubbles and foams are in the class of things, which few people ever notice or think about, but which appear in the daily lives of almost everyone in one way or another. This should not be surprising since we humans are entirely dependent on two of the most common and straightforward ingredients with which one can make foam and bubbles: air and water. Granted, using only air and water does not result in very long-lasting or complex foams, but it highlights the fact that foams can be found anywhere humans can live.

Other than just air and water, more complex foams can be created with many different mixtures of ingredients. Just by adding normal household soap, the created foam is already much longer-lasting and complex. If one traps this water-soap mixture inside a glass jar and mixes it well, it is possible to observe the evolution and coarsening of the resulting foam in real-time. This is one of the easiest and cleanest ways one can enjoy the intricacies of the evolution of a complex system. Other everyday situations where one encounters foams are the creamy froth of a *stein* of beer, the foamy pile of shaving cream, the dishwashing soap or the product one uses to wash one's hair. In other words, foams are everywhere in our daily lives.

Other than drinks and the hygiene industry, foams have bubbled up, if you pardon the pun, in many manufacturing, safety and protection related industries as a marvellous material. Applications of foams cover a plethora of diverse topics such as flotation, which is the process of separating different species of particles based on their hydrophobicity, used in mineral processing; fire-fighting foams, which are used to extinguish and prevent fires and to suppress reignition and hazardous vapors; blastwave mitigation, where protective barriers of aqueous foams are used to transfer the energy of a blast wave in to less destructive forms; insulation, where a certain volume is to be filled with material to prevent heat leakage [1], [2]. Additionally, foams are used in different capacities in e.g. automotive, athletics, medical and packaging industries [2]. To put it more succinctly, foams are everywhere in our modern lives.

The study of foams, like many other physical phenomena, can be done experimentally or theoretically. Some sort of mixture of the two is a computer simulation of a particular theoretical model. Many of the theoretical models of foams bear re-

semblance to other physical phenomena related to cellular networks and thus insights from one can sometimes be applied in the other. As if the role of foams in industry and otherwise was not already established well enough, the study of foams can help understanding e.g. the properties of cooling metals, where impurities "coarsen" as the metals cool down.

In this thesis, I present a GPU (graphics processing unit) implementation of a particular wet foam model. The structure of the thesis is such that we first perform a theoretical background check for the simulation code in chapter 2. In this chapter we encounter two different theories for modeling the behaviour of foams; one for wet foams proposed originally by Lifshitz, Slyozov and Wagner and another for dry foams proposed originally by von Neumann [3], [4], [5]. After these two theories, we look at earlier attempts at simulation of foams and compare some of the strengths and weaknesses of each approach. We conclude the chapter with a detailed explanation of the wet foam model used in this thesis first proposed by Durian, later modified by Gardiner, Dlugogorski and Jameson, with further modifications by Khakalo, Baumgarten, Tighe and Puisto and with final modifications by the author [6], [7], [8].

Once we have established a firm foothold in the theory of foams and bubbles, we enter the territory of implementation and structure of the code in chapter 3. Here we start with a technical discussion on the differences between CPU and GPU programming and mention how memory layout affects the performance of the simulation. After this, we grab a metaphorical scalpel and dissect the simulation to look at the details of its structure and how the theory is realized in the code. We look at how the program state is prepared for the simulation and how the simulation loop performs the integration of the two important equations we will have met in the chapter on the theory of foams.

Once the implementation and structure of the code is clear and no corner has been left unexamined, in chapter 4 we analyze the behaviour of a two-dimensional foam simulated with the presented code. The thesis finishes with a quick recap of the work done and presents some conclusions finally in chapter 5. Now let us see what theory tells us of the behaviour of foams.

2. Theory

This chapter gives a whirlwind tour of some important theories and numerical models related to foam physics. The chapter starts with some theoretical background, specifically by introducing the Lifshitz-Slyozov-Wagner (LSW) theory and von Neumann's law in section 2.1. After these two fundamental theories, several different models used in the numerical simulation of foams are introduced in section 2.2. Finally, once the aforementioned introductions are made, section 2.3 concludes the chapter by giving a detailed explanation of the model used in the implementation made for this thesis.

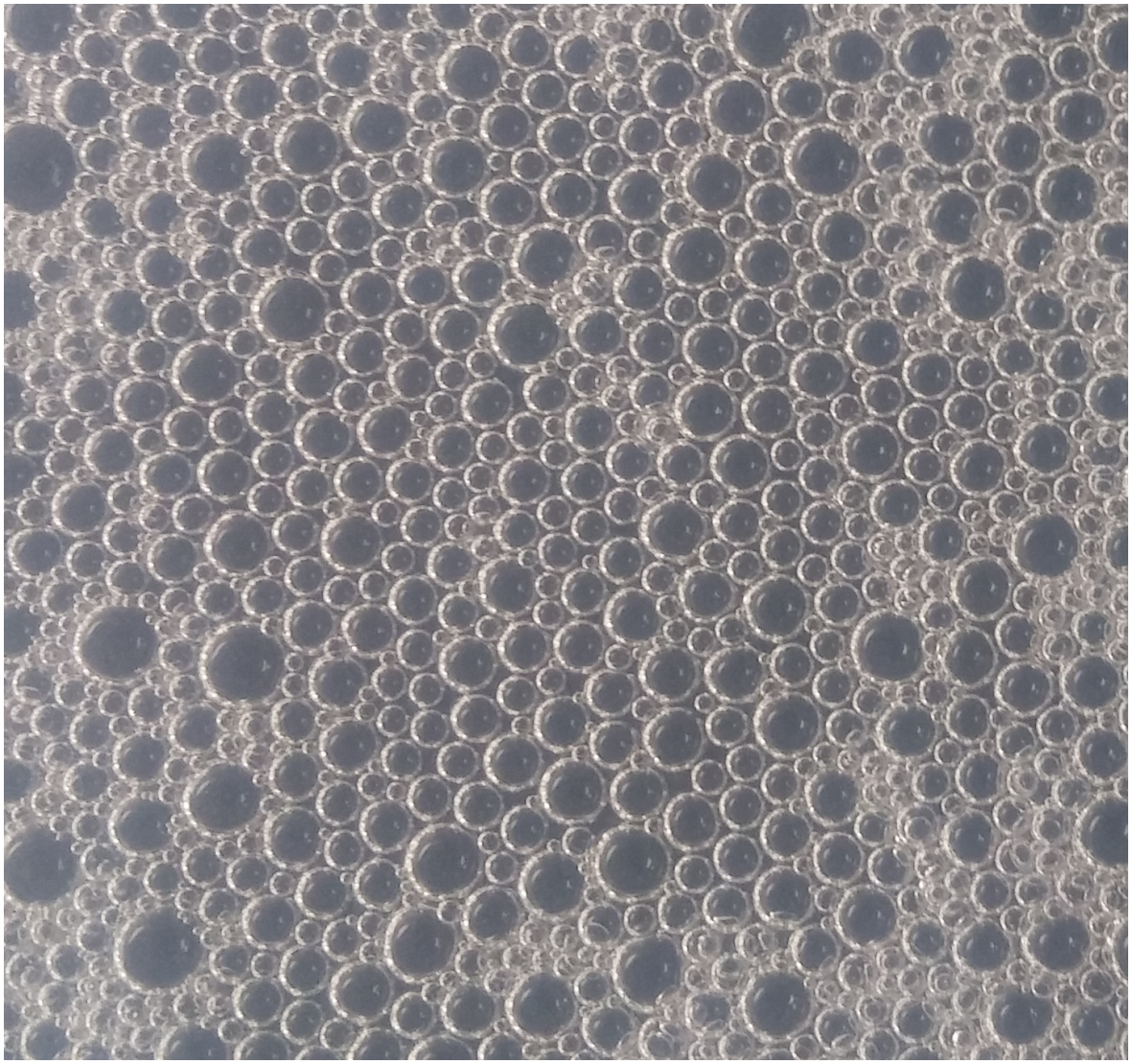
2.1 Theoretical background

For liquid foams, the diffusion of gas molecules through the thin film walls of bubbles is an important mechanism through which the foam tends to a thermodynamical equilibrium. Through diffusion, the bubbles with smaller than average size lose gas, while the bubbles of greater than average size gain gas. Following this process, the froth naturally evolves to a state, where the average size of bubbles steadily increases according to a power law. This diffusion process is broadly called coarsening and it is analogous to a phenomenon called Ostwald ripening, prevalent in emulsions and crystals. Thus, studying the coarsening of bubbly liquids and froths, insights into the evolution of crystals and other phenomena present themselves.

The evolution and structure of foams depend, among other things, on the wetness of the foam. In this chapter, we will look at two very different theories of foam evolution for two very different foams; one for very wet foams, another for very dry foams.

In the very wet limit, the froth is close to a bubbly liquid; the bubbles in the froth are spherical and no bubble is in contact with any other bubble as in figure 2.1. Thus direct bubble-to-bubble gas exchange cannot happen and the gas exchange is mediated entirely by a continuous liquid phase. The gas exchange in this scenario is dependent on the concentration of gas molecules in the liquid, which can be expressed as a gas pressure with the help of Henry's law. This gas pressure is then compared to the gas pressure inside a bubble, and if the pressure inside the bubble is larger than the gas pressure of the liquid medium, the bubble loses gas and shrinks. Conversely, if the

Figure 2.1: A photo of wet foam. Notice the spherical shape of individual bubbles and how the bubbles are barely apart from each other in the center of the picture. Picture own work by Juhana Lankinen (2019).



pressure inside the bubble is lower than the gas pressure of the medium, the bubble gains gas and expands. Seminal work on foam evolution in the very wet limit was done by Lifshitz & Slyozov (LS) and Wagner (W) [3][4]. Their work forms the basis for the so-called LSW theory and will be looked at in section 2.1.1.

While in the very wet limit the bubbles are spherical and the froth close to a bubbly liquid, in the very dry limit the situation is completely different. The drier the foam gets, the more the bubbles are distorted from their spherical shapes. In the very dry limit the foam consists of closely packed bubbles, the shapes of which are polyhedra in three and polygons in two dimensions. An example of a dry foam can be seen in figure 2.2. It is intuitively clear that the gas exchange between bubbles in a dry foam must be different from the gas exchange in a wet foam; after all, in a dry foam the bubbles are in direct contact with one another, their shapes are completely different, and there is very nearly zero liquid content to mediate the gas exchange. This is indeed the case as will be seen later in section 2.1.2, where von Neumann's theory of gas exchange for bubbles in a dry two-dimensional foam is introduced and explained. This theory was first formulated in [5] and it showed that the evolution of a bubble in a very dry two-dimensional foam is only dependent on the shape of the bubble, not its size or other attributes.

2.1.1 Lifshitz-Slyozov-Wagner theory of foam ripening

The LSW theory presented in this section closely follows the derivation in [1].

We start by considering a suspension of separated bubbles of radius R. The concentration of gas in the liquid $c = c(r, t)$ diffusing to and from bubbles in the liquid isotropically surrounds each bubble. The evolution of the concentration is described by Fick's second law of diffusion $\partial c/\partial t = D \cdot \Delta c$, where D is the diffusion coefficient of the particular gas in the liquid. Assuming that the time it takes for the gas concentration in the liquid to develop to a fully formed state is small compared to the time scale of the evolution of bubble radius, i.e. that $\partial c/\partial t \approx 0$, the concentration profile of the gas is expected to be

$$c(r) = c_\infty + R\frac{c(R) - c_\infty}{r}, \tag{2.1}$$

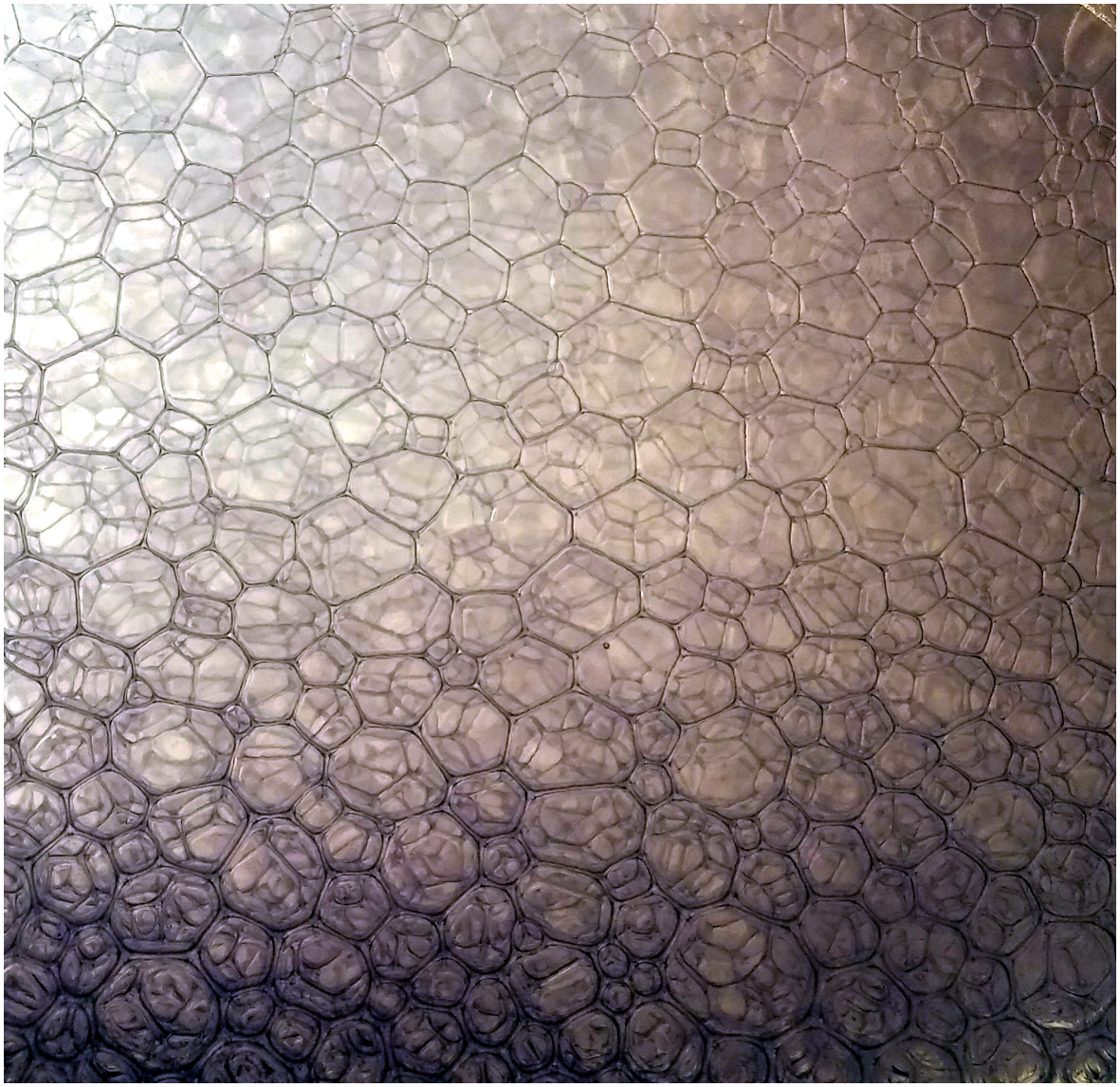
where c_∞ and $c(R)$ are the gas concentrations at $r = \infty$ and $r = R$, respectively.

The flux of gas across the surface of the bubble can be determined using Fick's first law

$$J = -D\frac{\partial c}{\partial r}, \tag{2.2}$$

from which the change of volume for a bubble can be found by multiplying the flux with the surface area of the bubble $4\pi R^2$ and the molar volume of ideal gas v_m such

Figure 2.2: A photo of three-dimensional dry foam. Notice the polyhedral shape of individual bubbles. Dryness increases upwards as gravity pulls the liquid downwards. Picture own work by Juhana Lankinen (2019).



that

$$\frac{dV}{dt} = -4\pi R^2 D v_m \frac{\partial c}{\partial r}_{|r=R} = 4\pi R^2 D v_m \frac{c_\infty - c(R)}{R} \tag{2.3}$$

The bubble gas pressure $P = P_0 + 2\sigma/R$ and the gas concentration at the surface of the bubble $c(R)$ are in equilibrium, and the gas pressure can be determined from Henry's law

$$c(R) = H(P_0 + \frac{2\sigma}{R}), \tag{2.4}$$

where H is Henry's constant, σ is the surface tension and P_0 a reference pressure.

Combining equations 2.3 and 2.4, the rate of change of radius for a bubble can be solved to be

$$\frac{dR}{dt} = \frac{1}{4\pi R^2}\frac{dV}{dt} = \frac{H D v_m P_0}{R}\left(s - \frac{2\sigma}{R P_0}\right), \tag{2.5}$$

where $s = (c_\infty - c_0)/c_0$ is the saturation parameter. From equation 2.5 it can be seen that for every value of $s > 0$ there exists a critical radius $R_s = 2\sigma/sP_0$ with which a bubble of gas is in equilibrium with the surrounding solution, i.e. $dR/dt = 0$. Both LS and W showed that $R_s = \langle R \rangle$, so equation 2.5 can be written more succinctly as

$$\frac{dR}{dt} = \frac{K}{R}\left(\frac{1}{\langle R \rangle} - \frac{1}{R}\right), \tag{2.6}$$

where $K = 2\sigma H D v_m$. From equation 2.6 it is readily seen that larger than average bubbles keep increasing in size, while smaller than average bubbles keep decreasing until they eventually disappear. This leads to the conclusion that the average size of the bubbles in the froth increases over time.

LS also showed that the growth of $\langle R(t) \rangle$ asymptotically approaches the form

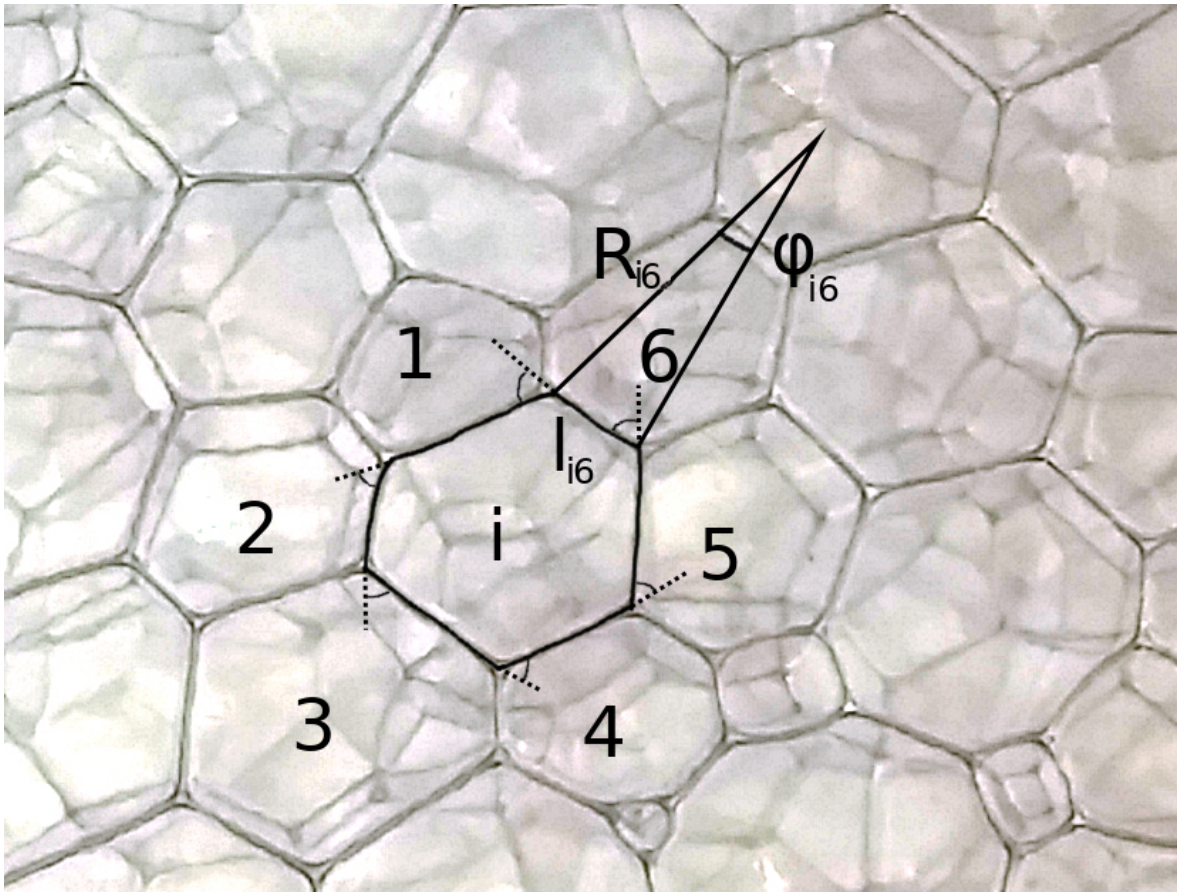
$$\langle R \rangle = \left(\frac{4}{9} K t^{1/3}\right) \tag{2.7}$$

This power law is applicable to dilute systems where the bubbles are not in contact with each other. In a froth with interacting bubbles, the power law does not hold in general. Next, we will see an example of a theory for bubbles in contact with each other, originally formulated by John von Neumann.

2.1.2 von Neumann's law

In 1952 John von Neumann did seminal work by showing that the evolution of individual bubbles in dry two-dimensional foams is only dependent on the shape of the bubble, not on its size. In this section I will go through the von Neumann theory and show his conclusion that the bubble's size will increase if it has more than six sides, decrease if it has less than six sides and stays unchanged if the number of sides is

Figure 2.3: An illustration showing the quantities related to von Neumann's law. Picture own work by Juhana Lankinen (2019), adapted from figure 4.2 in [1].



exactly six. Since von Neumann based his arguments on Plateau's laws of soap films, it is educational to review those before delving deeper to von Neumann's theory.

In 1873 Plateau published his findings on configuration and shape of foams based on his experimental observations. He formulated four laws that can be expressed as

1. Soap films consist of smooth surfaces.
2. These surfaces have a constant mean curvature across the film separating two bubbles.
3. Three soap films meet at 120° angles. The edge the three bubbles share is called a Plateau border.
4. Four Plateau borders meet at a vertex at $\cos^{-1}(-\frac{1}{3}) \approx 109.47°$ angles (the tetrahedral angle).

Let us now consider a two-dimensional foam that follows the laws of Plateau. In this type of foam, the edges of bubbles are spherical arcs of radius R_{ij} and adjacent arcs always meet at angles of $\frac{2\pi}{3}$. The gas flow across a thin film separating two bubbles is given by (see [1] for complete derivation)

$$\mathbf{q}_f = k_f(P_1 - P_2)\mathbf{e}_x, \tag{2.8}$$

where k_f is the permeability of the thin film, P_i are the pressures on the two sides of the film and $\mathbf{e}_x$ is a vector normal to the film. Using the Young-Laplace equation for the pressure difference $\Delta p = \frac{2\sigma}{R}$, l_{ij} for the length of the film separating the bubbles i and j and summing the gas flow across each film of a bubble we get an equation for the change in area of a two-dimensional bubble [1]

$$\frac{dA_i}{dt} = -2\sigma k_f \sum_j^n \frac{l_{ij}}{R_{ij}}, \tag{2.9}$$

where R_{ij} is the radius of curvature for the wall separating the bubbles under consideration. With the help of figure 2.3 we can define the ratio l_{ij}/R_{ij} as the angle ϕ_{ij}. Noting that the total curvature across the bubble must equal 2π we can find the amount of curvature that is left after we sum each of the n angles of $\pi/3$, which is the change in the tangent of a wall at each corner. The curvature that is left is

$$\sum_j^n \phi_{ij} = 2\pi - n\frac{\pi}{3} = \frac{\pi}{3}(6 - n) \tag{2.10}$$

Combining equations 2.9 and 2.10 we see that the change of area for bubble i is given by

$$\frac{dA_i}{dt} = \frac{2\pi}{3}\sigma k_f(n - 6) \tag{2.11}$$

So the size of the bubble increases if the number of sides is greater than 6, decreases if it is less than 6, and stays constant if it is exactly 6. Next, we will look at some numerical models used to study different properties of foams.

2.2 Overview of numerical foam models

This section gives brief overviews of several different numerical models used in simulating foams. Some of the models were originally developed for solving problems completely unrelated to foams, but have been generalized and modified to fit into this regime of physics, while others have been devised specifically with foams and bubbles in mind.

2.2.1 Potts model

The so-called Potts model has been used extensively in simulating many different kinds of phenomena [9]. Although it originally did not have anything to do with simulating foams, a generalized version of it has nonetheless been shown to be a valuable tool for studying the coarsening of bubbles and grain growth in metals.

One example of foam simulations using the Potts model was done in [10]. Their simulation uses a discrete lattice of sites, (pixels in 2D or voxels in 3D), each of which has a label. A region of sites with the same label constitutes one bubble, meaning the number of bubbles equals the number of unique labels among all the sites in the lattice. The simulation proceeds by a Monte Carlo procedure, similar to the Ising model, where a random site is chosen among all the sites. The site is then compared to a random neighboring site (four neighbors in 2D or 6 in 3D). If the chosen neighboring site has a different label, i.e. it belongs to a different bubble, it is checked if changing the label of the original site decreases the surface energy. If the energy is decreased, the new label is accepted. If the energy stays the same, the new label is accepted with 50% chance and if the energy increases, the new label is rejected.

As the simulation proceeds in the manner described above, the system tends to a state where the surface energy is minimized, which, in the case of foams, leads to coarsening of the foam, i.e. the average radius of the bubbles in the foams increases as the simulation proceeds.

The strength of the Potts model is that it is computationally inexpensive to simulate a large number of bubbles with it, compared to models where bubbles are simulated as particles interacting with each other in a continuous space.

A limitation of the Potts model has to do with its inherent discreteness. In a

foam, the time scale for the bubbles to readjust themselves under stress is much smaller compared to the time scale of the driving force of coarsening, which is the diffusion of gas across bubble walls. Thus, the walls of bubbles are close to minimal surfaces [11]. In the Potts model, the diffusion of particles across boundaries (the growth process) has a time scale comparable to the time scale of the diffusion of particles along the boundary (the process that causes changes in the shape of the boundary). This leads to a situation, where the formed boundaries can take shapes which are far from ideal minimal surfaces [12].

2.2.2 Surface evolver

The surface evolver is a simulation software developed by Brakke [13]. The software evolves a given input surface according to specified constraints and energy functions such that the given energy functions are minimized over the entire surface. In the case of surface tension, this leads to minimized surfaces, like soap films between boundaries or the entire structure of a froth.

The surface evolver can be used to study essentially any sort of problem, where the goal is to minimize a set of functions over a well-defined surface. Thus it is not constrained to bubble physics, although it has been used to study e.g. the viscosity of dry two-dimensional foams [14]. Overall its usefulness for bubble physics lies in the realm of dry foams.

2.2.3 Direct simulations

A loose family of models called direct simulations attempt to reproduce the equilibrium configuration of bubbles in a two-dimensional froth as it coarsens [15]. As the name implies, these models directly simulate the coarsening of the froth and include essential physics for gas diffusion and wall movement. These models decompose coarsening into two separate processes; a set of laws for the rate of change of area of a bubble (gas diffusion and wall movement) and a set of topological reconfiguration or scattering processes that occur when bubbles disappear.

As an example, Frost and Thompson simulated microstructural evolution in thin films [16]. They used a process in which the individual walls were approximated as circular arcs and the movement rate at a point on the wall was proportional to the local curvature of the wall at that point. The process involved a second step, where the location of a so-called triple point (Plateau border) was adjusted so that the three walls meeting at that point would meet at angles of 120°.

Other direct simulations have been studied in e.g. [17], [18] and [19].

2.2.4 Mean-field theories

Mean-field theories of foam coarsening embed the physical interactions of bubbles to a single function, typically of the form [15]

$$\frac{d\rho(n,a,t)}{dt} \propto \frac{da}{dt}(n-6)\rho(n,a,t), \tag{2.12}$$

As an example, Beenakker studied the evolution of a two-dimensional bubble froth on a topological network [20]. In this model, the bubbles were represented as vertices on a network with an area and an associated list of neighbors. The area is then evolved according to von Neumann's law and the bubbles are deleted and the topology rearranged according to the pre-defined scattering processes [15].

2.2.5 Vertex models

Another family of models called vertex models has been used to study two-dimensional dry foams [21] [22]. A basic idea in the vertex models is to reduce the number of degrees of freedom in complex systems as cellular networks and thus make long simulations more efficient and computationally inexpensive. These reductions are achieved by forcing the simulation into a two-dimensional plane and concentrating on the movement of the vertices at which three edges of bubbles meet (i.e. on the Plateau borders). These underlying assumptions (Plateau borders can be modelled as points and movement happens in two dimensions) also mean that the model is constrained to a regime of very dry two-dimensional froths.

2.3 Present model

In this section, the model used for this thesis is presented in detail. As the model is based on the bubble mechanics model by Durian, the section starts with a short introduction to the original model [8]. After this, the equations of motion governing the mechanical response of the foam are explained. The section ends with an explanation of the employed gas exchange model first introduced by Gardiner, Dlugogorski & Jameson (GDJ) and later modified by Khakalo, Baumgarten, Tighe & Puisto (KBTP) with further modifications made for this thesis [6] [7].

Before Durian's model, the simulations used to study foam mechanics were almost exclusively concentrated on dry two-dimensional foams consisting of a structure of polyhedral bubbles [8]. The goal of Durian's model was to introduce a procedure where the liquid content, dimensionality and randomness were easily variable. This was achieved by modeling individual bubbles as spheres, with no degrees of freedom

introduced for the bubble shapes. The liquid fraction could be varied by increasing or decreasing the size of the simulation box relative to the total volume (area in two dimensions) of the spherical bubbles. This was in contrast with the earlier models, but complementary in the sense that the earlier models tried to incorporate finite liquid fractions starting from the dry limit, while Durian's model attempted the same starting from the wet limit [8].

2.3.1 Equations of motion

The mechanical response of the foam is modelled with pairwise forces between neighboring bubbles in terms of the radii R_i and the positions $\mathbf{x}_i$ of the bubbles. The forces are separated into two components [8].

The first component models the repulsive force between bubbles originating physically from the energy cost of distorting spherical bubbles. If the distance between the centers of the two bubbles is greater than the sum of their radii, the repulsive force between the bubbles is zero. If the centers are closer than the sum of the radii, the force between the bubbles is modelled as two springs in series with the force being proportional to the overlap of the two bubbles i and j

$$\mathbf{F}_{ij}^{c} = \sigma_0 \langle R_{in} \rangle \left(\frac{R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j|}{R_i + R_j} \right) \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|}, \tag{2.13}$$

where σ_0 models the surface tension of the liquid and $\langle R_{in} \rangle$ is the average radius of the bubbles at the start of the simulation.

The second component is due to dissipation in the liquid between two bubbles, modelling a viscous drag

$$\mathbf{F}_{i}^{d} = -\mu_0 (\mathbf{v}_i - \langle \mathbf{v}_j \rangle), \tag{2.14}$$

where μ_0 is the viscosity of the fluid and $\langle \mathbf{v}_j \rangle$ is the velocity of the continuous liquid phase, commonly calculated as the average velocity of the neighboring bubbles. The model ignores inertial terms, as the masses of the individual bubbles are so small that the inertial terms are negligible compared to viscous terms (the system is overdamped). This means the sum of the compression force and the drag force must be equal to zero, i.e. $\mathbf{F}_i^c = -\mathbf{F}_i^d$, yielding

$$\frac{d\mathbf{x}_i}{dt} = \frac{\sigma_0}{\mu_0} \langle R_{in} \rangle \sum_{j}^{N} \mathbb{1}_C \left(\frac{R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j|}{R_i + R_j} \right) \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|} + \langle \mathbf{v}_j \rangle, \tag{2.15}$$

as the equation of motion for bubble i. $\mathbb{1}_C$ is an indicator function defined as

$$\mathbb{1}_C(j) \coloneqq \begin{cases} 1 & \text{if } j \in C, \\ 0 & \text{if } j \notin C \end{cases} \tag{2.16}$$

where C is the set of bubbles bubble i is in contact with.

The mechanical response of the entire foam can now be simulated in a molecular dynamics fashion, solving equation 2.15 for each bubble separately.

Note that the ratio μ_0/σ_0 works as a scaling factor for the dynamics of the system. The timescale of the simulation can be an arbitrary unitless number scaled with the ratio. In the simulations done for this thesis, the ratio was chosen to be unity, i.e. $\mu_0 = \sigma_0$.

2.3.2 Gas exchange

The original model by Durian was concentrated on the mechanical response of the foam, and time evolution of the bubble size distribution was not allowed [8]. Thus, the model was suitable only for the study of the rheological attributes of foams. In light of this, GDJ introduced a model for gas exchange between neighboring bubbles [6]. Additionally, GDJ enabled so-called rattlers – small bubbles with no neighbors – to participate in gas exchange by comparing their internal pressure to the gas pressure (gas concentration) of the continuous liquid phase and changing their size accordingly.

KBTP modified this model by adding a mean-field term for gas exchange so that all the bubbles in the foam, not just the rattlers, would exchange gas with the liquid phase [7]. The model used in this thesis further refines this continuous phase gas exchange term by considering the free surface area of each bubble, i.e. the surface area that is not in contact with any other bubble. What follows is a detailed explanation of the gas exchange employed in the current model.

The total gas exchange is a combination of two different processes, namely gas exchange through thin films $\dot{V}_i^{tf}$ and gas exchange mediated by the continuous fluid phase $\dot{V}_i^{fp}$. The total gas exchange of each bubble is then the sum of these two distinct terms

$$\dot{V}_i = \dot{V}_i^{tf} + \dot{V}_i^{fp}, \tag{2.17}$$

where the dot notation denotes a time derivative ($\dot{x} = \frac{dx}{dt}$) as is commonly done in physics literature. The final rate of change for the radius of bubble i is calculated from the rate of change of volume

$$\frac{dV}{dt} = \frac{dV}{dR}\frac{dR}{dt} \rightarrow \frac{dR}{dt} = \frac{1}{A}\frac{dV}{dt}, \tag{2.18}$$

where the chain rule was invoked. In the next section each of these gas exchange processes are explained in detail, starting from gas exchange through thin films.

Through thin films

Gas exchange through thin films happens between two bubbles that are overlapping each other. Figure 2.4 illustrates the situation. The following derivation for the rate of change of volume for a bubble closely follows that presented in [23] and the final form is unchanged from that of either GDJ or KBTP.

The difference in pressure between two bubbles is given by the Laplace-Young equation

$$\Delta P_{ij} = 2\gamma \left(\frac{1}{R_i} - \frac{1}{R_j} \right), \tag{2.19}$$

where γ is the surface tension and R_i, R_j are the radii of the two (spherical) bubbles. The molar mass transfer rate of gas is proportional to the area through which the transfer takes place and the pressure difference between the bubbles

$$\frac{dn}{dt} = -JA\Delta P, \tag{2.20}$$

where J is the overall mass transfer coefficient or the effective permeability to mass transfer. Combining the equations 2.19 and 2.20 leads to

$$\frac{dn_{ij}}{dt} = 2\gamma JA \left(\frac{1}{R_j} - \frac{1}{R_i} \right) \tag{2.21}$$

As the excess pressure inside a bubble and the pressure difference between two bubbles is small relative to the atmospheric pressure P_a, the molar mass transfer rate can be expressed as the rate of change of volume by applying the ideal gas law

$$\frac{dn}{dt} = \frac{dn}{dV}\frac{dV}{dt} \rightarrow \frac{dV}{dt} = \frac{\mathbb{R}T}{P_a}\frac{dn}{dt}, \tag{2.22}$$

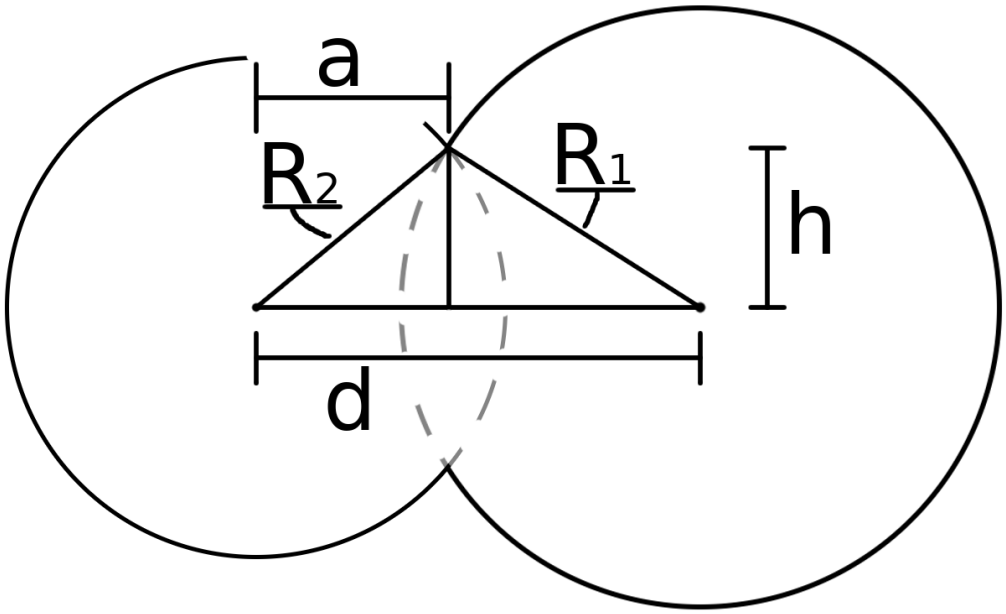
where $\mathbb{R}$ is the gas constant and T is the temperature. This leads to a final form for the rate of change of volume through thin films between a bubble i and its neighbors j

$$\dot{V}_i^{tf} = K \sum_j^N \mathbb{1}_C A_{ij} \left(\frac{1}{R_j} - \frac{1}{R_i} \right), \tag{2.23}$$

where $K = \frac{2\gamma \mathbb{R}TJ}{P_a}$ and A_{ij} is the area of overlap between bubbles i and j.

As mentioned earlier in section 2.3.1, the dynamics of the bubble interaction is scaled by the ratio μ_0/σ_0. To prevent the bubbles from growing so fast that they start to overlap each other significantly, it is advantageous to have the mechanical response of the froth to be significantly faster than the rate of gas exchange in the simulation. This reflects the real behaviour of froth under tension, where the process of readjustment has significantly smaller time scale compared to the process of coarsening, as already discussed in section 2.2.1. With this in mind and by noting that the unit of the

Figure 2.4: Overlapping bubbles



parameter K is $Length^2/time$, the time scale of the simulation is non-dimensionalized as: $t^* = \frac{Kt}{\langle R_{in}\rangle^2}$. The parameter K is chosen so that the ratio $\langle R_{in}\rangle^2/K$ is large compared to the ratio μ_0/σ_0.

The only thing missing from the equation for gas exchange through thin films is the surface area for the gas transfer or the area of overlap between two bubbles. The area can be calculated with the help of figure 2.4. After some arithmetical trickery, the quantity h can be shown to be

$$h = \frac{1}{2d}\sqrt{4d^2R_2^2 - d^4 + (R_1^2 - R_2^2)^2 - 2d^2(R_1^2 - R_2^2)}, \tag{2.24}$$

where d is the distance between the bubbles and R_1, R_2 are the radii. From figure 2.4 it is quickly seen, that the area of overlap is πh^2 in three dimensions and $2h$ in two dimensions. A short glance at equation 2.23 readily assures us* that the total volume of the bubbles stays constant, i.e. $\sum_i^N \dot{V}_i = 0$, so the gas fraction is conserved by the gas exchange through thin films.

*$\dot{V}_{ij} = -\dot{V}_{ji}$

Through medium

The second process with which the bubbles exchange gas with each other is mediated by the continuous fluid phase. GDJ used this process only with bubbles that were not in contact with any other bubble. They redistributed the gas content of the liquid phase according to the radius of each bubble, so that the total volume of gas would be conserved. The final form of gas exchange they used for these bubbles was

$$\frac{dR_i}{dt} = K\left(\left\langle \frac{1}{R_j} \right\rangle - \frac{1}{R_i}\right), \tag{2.25}$$

where the average is notably of the reciprocal of radii, which does not conserve the volume, and thus the need for "artificial" redistribution of gas arises.

In their version, KBTP expanded this idea to allow all the bubbles in the simulation to take part in the gas exchange through the fluid phase, not just the rattlers. They introduced a κ parameter to control the strength of $\dot{V}^{fp}$ in the overall gas transfer, changed the term each bubble is compared to from the average of reciprocal $\langle \frac{1}{R} \rangle$ to the reciprocal of average $\frac{1}{\langle R \rangle}$ and added an ad-hoc term of $1-\phi$, where ϕ is the gas fraction of the froth, based on the reasonable requirement that the gas exchange mediated by the liquid phase should go to zero as the liquid phase disappears [7]. The final form of gas exchange mediated by the liquid phase used by KBTP is

$$\dot{V}_i^{fp} = 2\pi\langle R_{in}\rangle(1-\phi)\kappa K\left(\frac{1}{\langle R\rangle} - \frac{1}{R_i}\right) \tag{2.26}$$

The fluid phase gas exchange used in this thesis is similar to that of KBTP but slightly refined. The ad-hoc term of $1-\phi$ is replaced by the ratio of free surface area of a bubble to the average surface area of all bubbles $\frac{A_i^f}{\langle A\rangle}$, where $A_i^f = A_i - \sum_j^N \mathbb{1}_C A_{ij}$. This term naturally tends to zero as the gas fraction of the foam is increased, as a greater area of the bubble's surface area will be in touch with neighboring bubbles. It is noteworthy, that since the simulated bubbles are always considered to be spherical as per the original assumption of Durian [8] and bubbles will never deform regardless of the dryness of the froth, the term will never actually reach zero. That should not be of great concern, since there are many more suitable models for the simulation of very dry foams with polyhedral bubbles, and limited liquid content in any case.

Another change in the fluid phase gas exchange used in this thesis arises from the usage of the free surface area as a scaling factor for individual bubbles. The term each bubble is compared to has to be changed from the reciprocal of the average radius to conserve the total volume of gas. With this in mind, the final form of the gas exchange mediated by the continuous liquid phase used in this thesis is

$$\dot{V}_i^{fp} = \kappa K\langle A_{in}\rangle\frac{A_i^f}{\langle A\rangle}\left(\frac{1}{\rho} - \frac{1}{R_i}\right), \tag{2.27}$$

where $\langle A_{in} \rangle$ is kept in the equation to preserve correct dimensions.

From this equation the ρ term can be solved with the requirement that the volume is conserved, i.e. that $\sum_i^N \dot{V}_i^{fp} = 0$ and with the assumption that ρ is the same for all bubbles

$$\sum_i^N A_i^f \left(\frac{1}{\rho} - \frac{1}{R_i} \right) = 0 \rightarrow \frac{1}{\rho} \sum_i^N A_i^f - \sum_i^N A_i^f \frac{1}{R_i} = 0 \rightarrow \rho = \frac{\sum_i^N A_i^f}{\sum_i^N A_i^f \frac{1}{R_i}} \tag{2.28}$$

Final rate of change of radius

The work finally bears fruit once we gather the relevant equations from the last few sections and combine them to produce the equation for the final rate of change for the radius of bubble i. Specifically, combining the equations 2.17, 2.18, 2.23 and 2.27, we get

$$\dot{R}_i = \frac{K}{A_i} \left(\sum_j^N \mathbb{1}_C A_{ij} \left(\frac{1}{R_j} - \frac{1}{R_i} \right) + \kappa \langle A_{in} \rangle \frac{A_i^f}{\langle A \rangle} \left(\frac{1}{\rho} - \frac{1}{R_i} \right) \right) \tag{2.29}$$

as the final form. This equation and the equation of motion (2.15) shown earlier form the basis for the simulations done in this thesis. How exactly these equations are solved and used to simulate the mechanics and coarsening of froth are questions to be answered in the next chapter, where the details of the simulations and the implementation of the theory are explained in great detail.

3. Implementation

The focus of this chapter is the implementation of the previously introduced theory. We start the chapter in section 3.1 with a short discussion on how the simulation data is stored in the GPU memory and why this is important. We then go through a high-level overview of the structure of the program in section 3.2. In section 3.3, which is the last section of the chapter, I discuss arguably the most important part of simulation; the simulation loop itself. In this section, we scrutinize the details of the simulation and go through sections like the integration loop, bubble removal and neighbor search.

3.1 Memory layout

Traditionally in object-oriented CPU programming, some sort of a structure is used as a wrapper around related pieces of data. If the program uses multiple instances of this structure, the different structures are usually stored in an array of some kind. The implementation of the structure and the array of structures might be similar to the example implementation shown in listing 3.1.

Listing 3.1: Bubble structure

```
struct Bubble
{
    double x = 0;
    double y = 0;
    double z = 0;
    double r = 0;
}

Array<Bubble> bubbles;
```

The data usually ends up being laid out in memory in such a way that the individual pieces of data (i.e. the member variables of the structure) are stored continuously, such that the entire contents of one structure are before the entire contents of the next

Figure 3.1: Memory layout for an array of structures

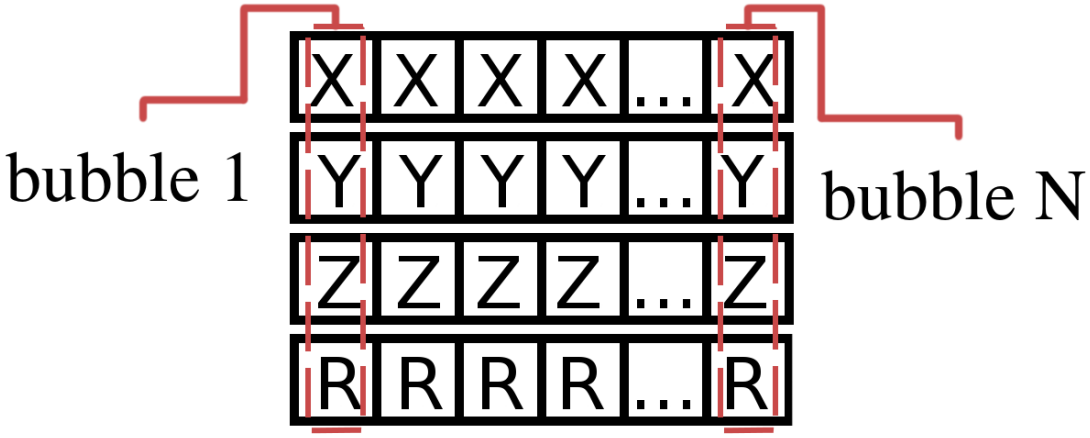
structure, as illustrated in figure 3.1.

This style of memory layout is usually efficient for serial programs where a single processor or thread performs the operations in a loop. This is because the way of processing data in this kind of serial program many times follows a pattern, where an object or an interrelated bunch of data is first accessed, then all the specified operations are performed on that data, and finally the loop counter is incremented and the next object or bunch of data receives the same treatment. Having all the related bits of data close to each other in memory increases the performance of this sort of processing, as a larger portion of the cache line or block is filled with useful data, thus reducing the chances of cache misses. If the array of structures is also ordered such that the objects that are processed *temporally* close to each other are also close to each other in memory and the structures are relatively small, one cache line might contain enough data for a few loops, thus reducing cache misses even more and leading to efficient use of the computation resources. More info on efficient memory layouts in serial programming and other related topics can be found from e.g. [24] and the references therein.

In parallel computing, however, storing data as an array of structures is usually a bad idea. Many times different threads – to use the CUDA terminology – access different data and perform operations separate from, but at the same time with, each other. This is a fundamentally different way of performing operations from the way serial processing is done. Instead of a single processor or a thread doing work on a single bunch of data at a time, tens, hundreds or thousands of threads might be performing the same operations on a different bit of data at the same time. The only way this can be efficient is if the memory accesses of each of these threads or processors can be satisfied with a few memory transactions.

Generally speaking, the more scattered the accessed memory addresses across the threads within a CUDA warp are, the more the throughput is usually reduced. If all the threads of a warp wanted to read the member variable x of the particular bubble belonging to the thread and the memory was saved as illustrated in figure 3.1, the

Figure 3.2: Memory layout for a structure of arrays



throughput would be greatly reduced as $\frac{3}{4}$ of the memory is polluted with completely unnecessary bits of data (y, z and r in this case). The same thing would repeat when each thread wanted to access y, and again with z and r.

If, however, the memory was saved as a structure of arrays, as listing 3.2 and figure 3.2 show, the individual memory accesses of the threads would be coalesced into fewer memory transactions, thus increasing the throughput. This is because more useful data would fit into the same space, thus allowing more threads to benefit from a single memory transaction. This is, in fact, the way the data is stored by the simulation code implemented for this thesis.

More details of memory access patterns in CUDA and maximizing memory throughput can be found in [25]. Now, let us turn our attention to the structure of the program.

Listing 3.2: Bubble arrays

```
struct BubbleStruct
{
    Array<double> x;
    Array<double> y;
    Array<double> z;
    Array<double> r;
}

BubbleStruct bubbles;
```

3.2 Structure of the program

LISTING 3.3: Structure of the program

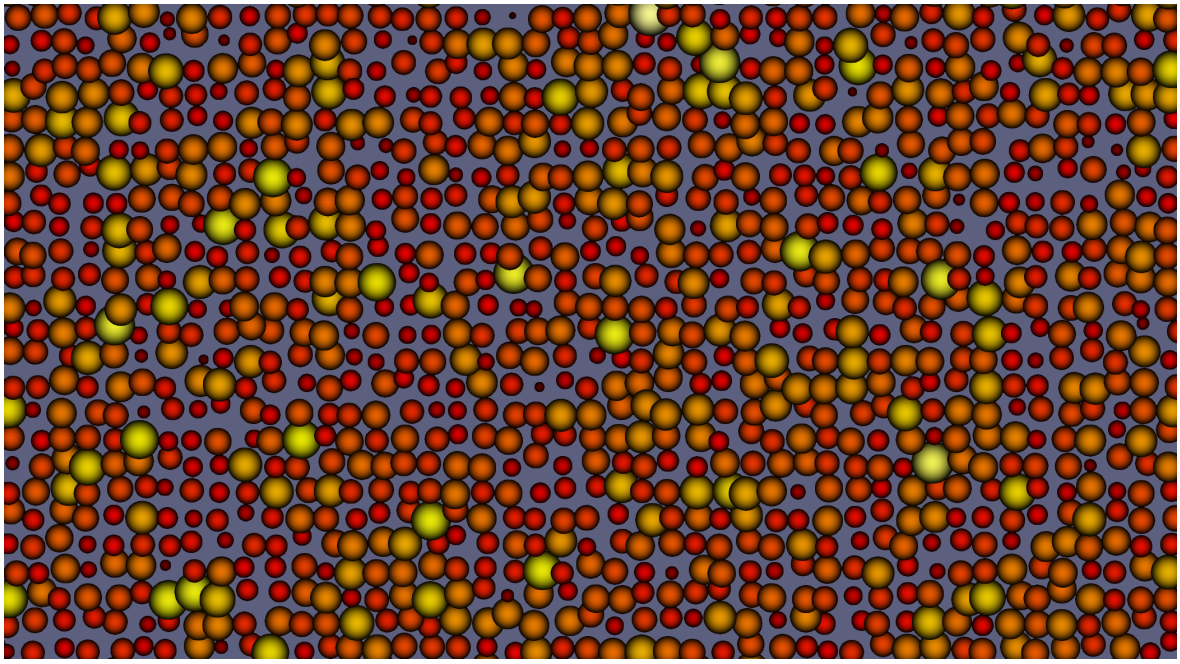
```
readInputParameters();
setupSimulation();
generateBubbles();
scaleSimulationBox();
stabilizeSimulation();

// Begin main simulation loop, coninue until end condition is met
// This is where most of the simulation time is spent
while (endConditionMet() == false)
{
    integrate();

    if (numberOfBubblesBelowMinRadius > 0)
        removeBubbles();

    // Not updated every time
    if (neighborsNeedUpdating())
        updateNeighbors();
}
```

The program consists of four main parts: setup, scaling, stabilization and finally simulation loop. In this section I explain the implementation behind the first three; the last and the most interesting of the four parts, simulation loop, is looked at in detail in section 3.3. A simplified example of the structure of the program is given in pseudocode in listing 3.3.

3.2.1 Setup

At the start of the program, the simulator and the simulation environment are constructed. The program reads inputs from a *.json* file and constructs a simulation environment from these inputs (see appendix A.1 for a full list of inputs). The simulation environment is then passed to the simulator, which uses the environment to set its internal variables. Device memory is also allocated at this point. A few large allocations are made at the start of the simulation and the memory stays allocated

Figure 3.3: The bubbles after data generation at random positions. The original grid-like structure is still clearly visible. Color coding is relative to bubble radius.



until the program is terminated. CUDA specific setup is handled after the allocations, including stream and event creation and symbol address fetching.

After the simulator and the environment have been constructed, the initial data for each bubble is created and stored in the allocated device memory. The radii of the bubbles are drawn from a Gaussian distribution, for which the mean and the standard deviation are given to the simulation as input parameters.

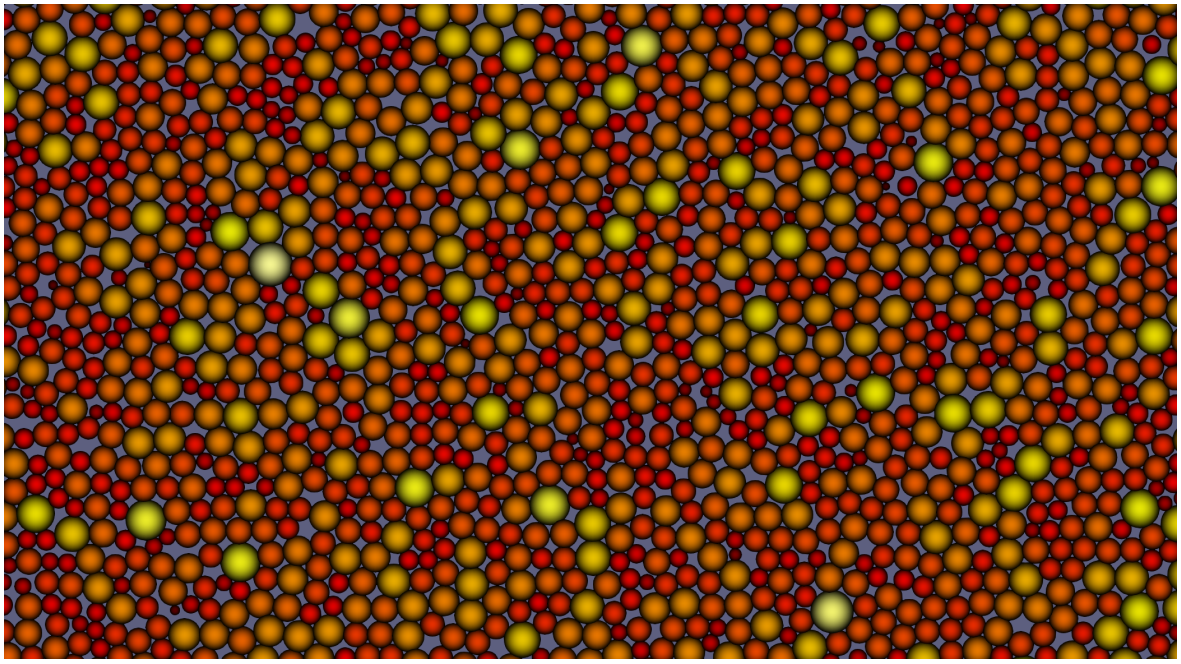
The positions of the bubbles are generated in a slightly more advanced manner. First, the entire simulation box is divided evenly to a grid of points. The number of points in each dimension is roughly the size of the simulation box in that dimension divided by the average radius of the bubbles. Then each of these evenly distributed grid points is offset by a random vector. These offset points are the starting positions of the bubbles. With this scheme, the bubbles end up roughly evenly distributed in space but with random starting positions nonetheless.

All the other data for each bubble is generated from the radius and the positions. Initial velocities are calculated from the positions of the bubbles as per the equation 2.15. These velocities are then used to calculate new positions using the well known Euler method

$$y_{n+1} = y_n + hf(t_n, y_n), \tag{3.1}$$

where h is a chosen time step, $t_n = t_0 + nh$ and $f(t, y) = y'$. Note that the Euler method is only ever used once and only once during the entire simulation and it is at this point. This is because the Adams-Bashforth-Moulton predictor-corrector integration

Figure 3.4: The bubbles of figure 3.3 after scaling the simulation box and after a few initial integration steps. The system and the gas fraction are the same.



scheme needs some initial values as a kickstart. Once the simulation has been set up, a snapshot of the current configuration of bubbles is saved, and the scaling stage begins.

3.2.2 Scaling

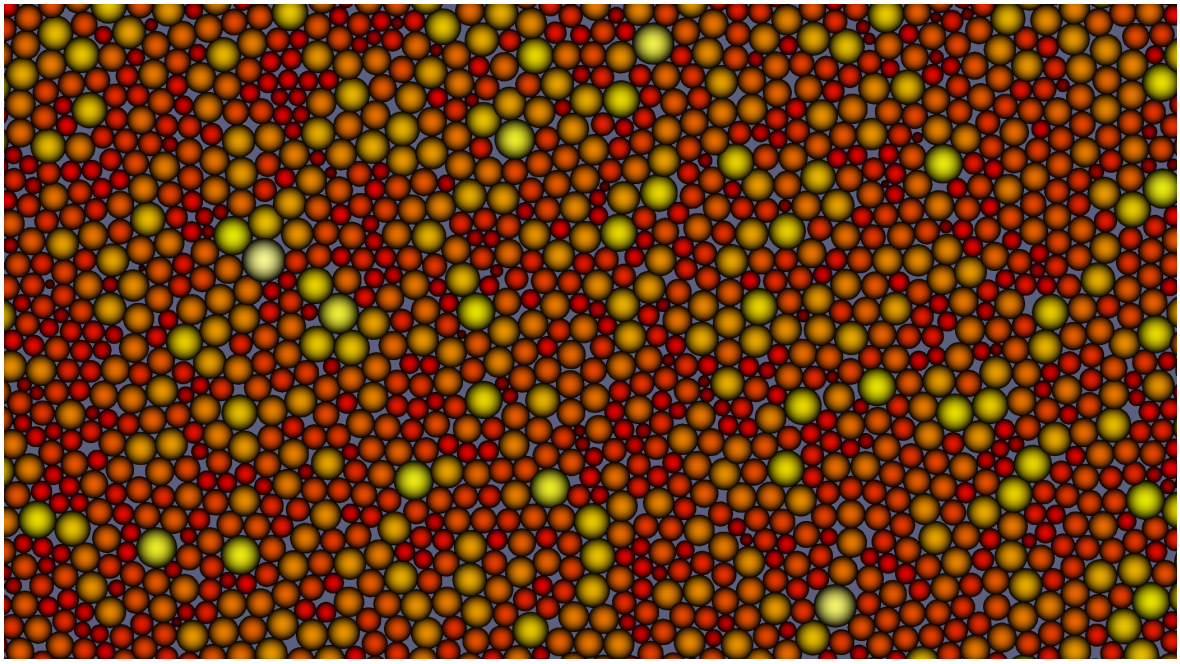
In scaling stage* the simulation box is scaled according to the desired gas fraction. If the generated bubbles fill more volume than the target gas fraction, the simulation box is scaled up. Conversely, if the target gas fraction is greater than the relative volume filled by the bubbles, the simulation box is scaled down. Before scaling, the positions of the bubbles are transformed into normalized coordinates of the simulation box. Then the entire box is scaled up or down, after which the normalized coordinates are transformed back to absolute positions. This way the amount of space between bubbles is changed uniformly everywhere inside the simulation box.

3.2.3 Stabilization

After the simulation box has been scaled properly so that the bubbles fill the desired fraction of the box, the stabilization stage is begun. During stabilization, the bubbles are allowed to move, but not change in size. In other words, the bubbles interact according to their equation of motion, but gas exchange is turned completely off.

*__N.B.__ This should not be confused with the *scaling state*, which describes the physical state of the foam at which point the radii of the bubbles grow according to a power law.

Figure 3.5: The same system of bubbles as in figures 3.3 and 3.4, but after stabilization. Notice the much closer packing relative to figure 3.4.



Since the bubble-bubble interaction is modelled as two springs in a series, the potential energy of the entire system is calculated as the energy stored in all the compressed bubbles (springs)

$$E^p_{tot} = \frac{1}{2}\sigma_0 \sum_i^N \sum_j^N \mathbb{1}_C \left(R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j|\right)^2, \tag{3.2}$$

where the symbols are as explained in section 2.3.1.

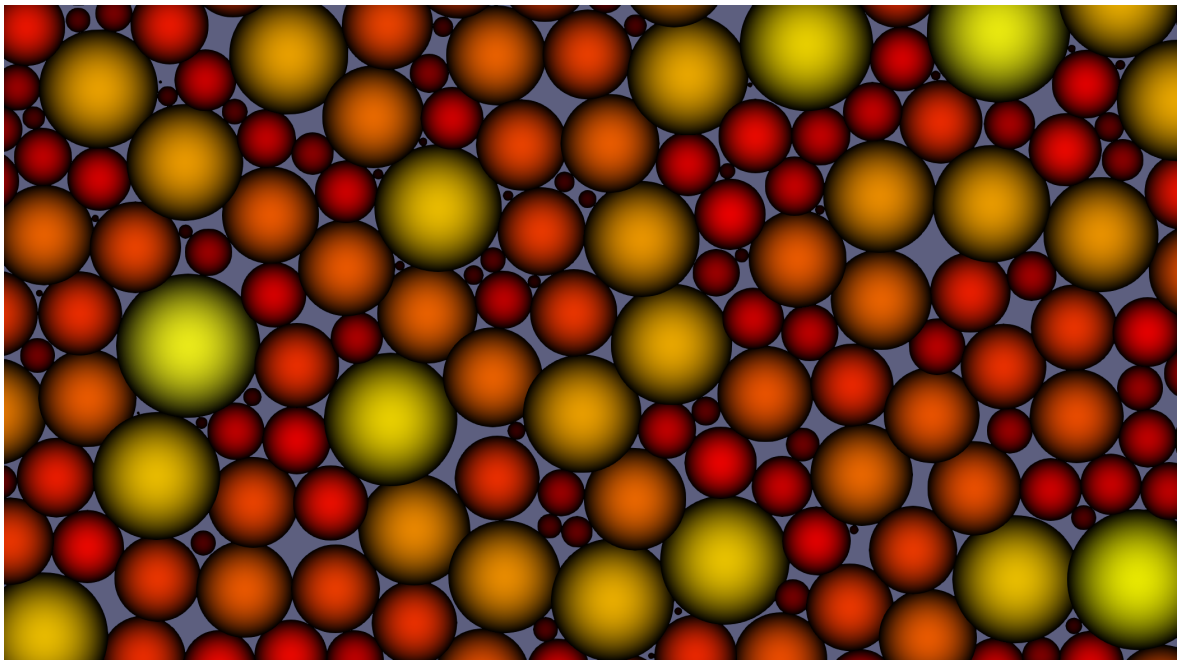
The stabilization runs in a loop, at the start of which the current energy is calculated. Then the system is allowed to relax (i.e. the equations of motion are solved) for a specified number of steps, after which the energy of the system is calculated again. If the change in energy is greater than a specified limit, the loop continues. If the change is less, the loop is terminated and the stabilization stage ends. At this point, a snapshot of the system is saved and the simulation proper starts.

3.3 Simulation loop

After all the preparations have been completed, the simulation is ready to start. In this section, I will first give an overview of the simulation loop and later we will look at the implementation of each part of the simulation loop in detail.

The simulation loop runs for most of the duration of the entire simulation. It has two end conditions, either of which immediately ends the simulation. After the

Figure 3.6: The bubbles after the simulation loop has been running for a while. Notice the great difference between the sizes of the the largest and the smallest bubbles compared to the setup in figure 3.3.



simulation stops, a final snapshot of the simulation is saved and all the allocated resources are freed. The two possible end conditions of the simulation loop are:

1. Number of bubbles left in the simulation is equal to or less than a pre-defined value.

2. The diameter of the largest bubble is equal to or greater than the minimum dimension of the simulation box.

The predefined value mentioned in end condition 1 is given as an input parameter to the simulation. Due to implementation details, the value is roughly 300 in two and 900 in three dimensions.

The first thing that happens during every execution of the simulation loop is the integration step. This step is itself a loop, which might run multiple times, but is always executed at least once. The integration step first produces a prediction of positions and radii, based on the current and previous rates of change. After the prediction, new, predicted rates of change are calculated based on the predicted positions and radii. Finally, all these predictions are used to calculate corrected positions and radii for each bubble. This scheme is called the Adams-Bashforth-Moulton prediction-correction scheme, where the explicit Adams-Bashforth method is used to calculate a new predicted value, which is then used by the implicit Adams-Moulton method to cor-

rect that prediction. Generally, these types of methods are known as linear multistep methods.

If the absolute difference between any of the predicted and corrected values is larger than a predefined value, the time step of the integration step is decreased and the integration step is repeated. If, on the other hand, the error or difference between the predicted and corrected values was sufficiently small, the time step is increased for future steps. In this way, the time step adapts during the simulation.

If any of the bubbles have shrunk below a certain size limit during the integration step, the gas content (volume) of those bubbles is distributed among the surviving bubbles and the small bubbles are removed. If either any bubbles were removed or a certain amount of integration steps has been taken, the neighbor list of each bubble is updated. During neighbor search the simulation box is divided into smaller rectangular cells, the number of which is relative to the number of bubbles left in the simulation. The information regarding which bubbles are neighbors with each other is saved as index pairs. Instead of a list of tuples, the indices are saved as two separate lists of indices for memory access coalescing reasons, as explained in section 3.1.

Once the integration step, bubble removal and neighbor search have all been completed, one step of the simulation loop is over and the loop restarts from the top. Next, we will take a detailed look at each specific step of the simulation loop. We start first with the integration step.

3.3.1 Integration loop

The first part of the simulation loop is the integration step. This part is explained in the following section. A simplified example of the entire integration loop is given in listing 3.4.

Prediction step

A second-order Adams-Bashforth method is used at the prediction step to generate an approximate value for each of the position coordinates and the radius. Given previous values, the method solves the next value with

$$\hat{y}_{i+1} = y_i + h[\frac{3}{2}f(t_i, y_i) - \frac{1}{2}f(t_{i-1}, y_{i-1})], \tag{3.3}$$

where h is the time step of the simulation, y is one of the positions coordinates or radius and $f(t_i, y_i) \equiv f_i$ is the velocity or rate of change for a given y. The predicted values $\hat{y}_{i+1}$ are then used in calculating new velocities in each dimension and new rates of change for radii as specified in equations 2.15 and 2.29, respectively.

LISTING 3.4: Integration loop

```
do
{
    predict();
    calculateVelocities();
    // During stabilization radii are held constant
    // and this step is skipped.
    calculateRateOfChangeForRadius();
    correct();

    difference = getDifference();
    if (difference > maximumDifference)
        decreaseTimeStep();
    else if (difference << maximumDifference)
        increaseTimeStep();
}
while (difference > maximumDifference)
```

Correction step

A second-order Adams-Moulton method is used at the correction step to calculate new values for the position coordinates and the radius based on the velocities and rates of change computed with the predicted values. Given previous values, the method solves the next value using

$$y_{i+1} = y_i + \frac{h}{2}[f(t_{i+1}, \hat{y}_{i+1}) + f(t_i, y_i)], \tag{3.4}$$

where the values are as explained below equation 3.3 but the value of the function f_{i+1} is the velocity or rate of change solved using the predicted value $\hat{y}_{i+1}$. Equation 3.4 is more commonly known as the *trapezoidal rule*.

Difference and adaptive time step

Once the correction step is completed, the difference between the predicted and the corrected values is taken simply as

$$d = max(d_j), \tag{3.5}$$

where

$$d_j = max(|y_{i+1,j,k} - \hat{y}_{i+1,j,k}|), \tag{3.6}$$

where the subscript j goes over the position coordinates and radius and the subscript k goes over each bubble.

The difference is then compared to a predefined maximum difference given as input argument. If the calculated difference is smaller than the predefined maximum difference, the time step of the integration is increased by a factor of 1.9 and the integration loop exits. On the other hand, if the difference is equal to or greater than the maximum allowed difference, the time step is decreased by a factor of 0.5 and the entire integration loop is repeated, starting again from the prediction step.

3.3.2 Bubble removal

During bubble removal, bubbles whose radius is less than a specified minimum radius are removed from the simulation, thus reducing the number of simulated bubbles. The minimum radius is set to be 10% of the input average radius $\langle R_{in} \rangle$. To conserve the gas fraction i.e. the total volume of gas in the simulation, the gas of the removed bubbles needs to be redistributed among the surviving bubbles such that

$$\sum_i^N V_i^{new} = \sum_j^M V_j^{old} = \sum_i^N V_i^{old} + V_{excess}, \tag{3.7}$$

with $M > N$. The fraction of the excess gas V_{excess} each surviving bubble receives is proportional to the bubble's current volume such that $V_i^{new} = V_i^{old} + aV_i^{old} = V_i^{old}(1+a)$. Combining this with equation 3.7 we get

$$\sum_i^N V_i^{new} = (1+a)\sum_i^N V_i^{old} = \sum_i^N V_i^{old} + V_{excess} \tag{3.8}$$

from which the multiplier can be solved to be

$$a = \frac{V_{excess}}{\sum_i^N V_i^{old}} \tag{3.9}$$

Using this multiplier the increase in radius of a bubble is then $R_i^{old}\sqrt{a}$ in two and $R_i^{old}\sqrt[3]{a}$ in three dimensions for each surviving bubble.

3.3.3 Neighbor search

The neighbors of each bubble are updated any time a bubble is removed from the simulation or if a certain amount of simulation steps has been taken after the last update. At the start of the search, the entire simulation box is divided into smaller rectangular cells. Each bubble is then assigned to a particular cell based on the position of the bubble.

After the cell division, the bubbles are reorganized in memory such that the bubbles of the first cell are stored first in the memory, the bubbles of the second cell are stored next and so on. This has the benefit that bubbles that are spatially close to each other are also close to each other in memory, thus reducing the number of needed memory transactions for the next stages for reasons explained in section 3.1.

Once the bubbles have been reorganized, the neighbor search is begun. The simulation box has been divided into equisized cells earlier and the neighbors of each bubble are searched from its cell as well as the adjacent cells. This is done in a manner where each cell is responsible for finding the neighbors to its bubbles from a "pool" of bubbles that contains all the bubbles in the cell itself but also the bubbles in its lower, left and back neighbor cells. In two dimensions this means that each cell compares its bubbles to every other bubble within itself but also every bubble in the western, southwestern, southern and southeastern cells. Figure 3.7 illustrates this.

If a bubble b_i in cell c_x is found to be the neighbor of bubble b_j in cell c_y, the indices i and j are compared and the smaller index is stored in index list 1 and the larger index is stored in index list 2.

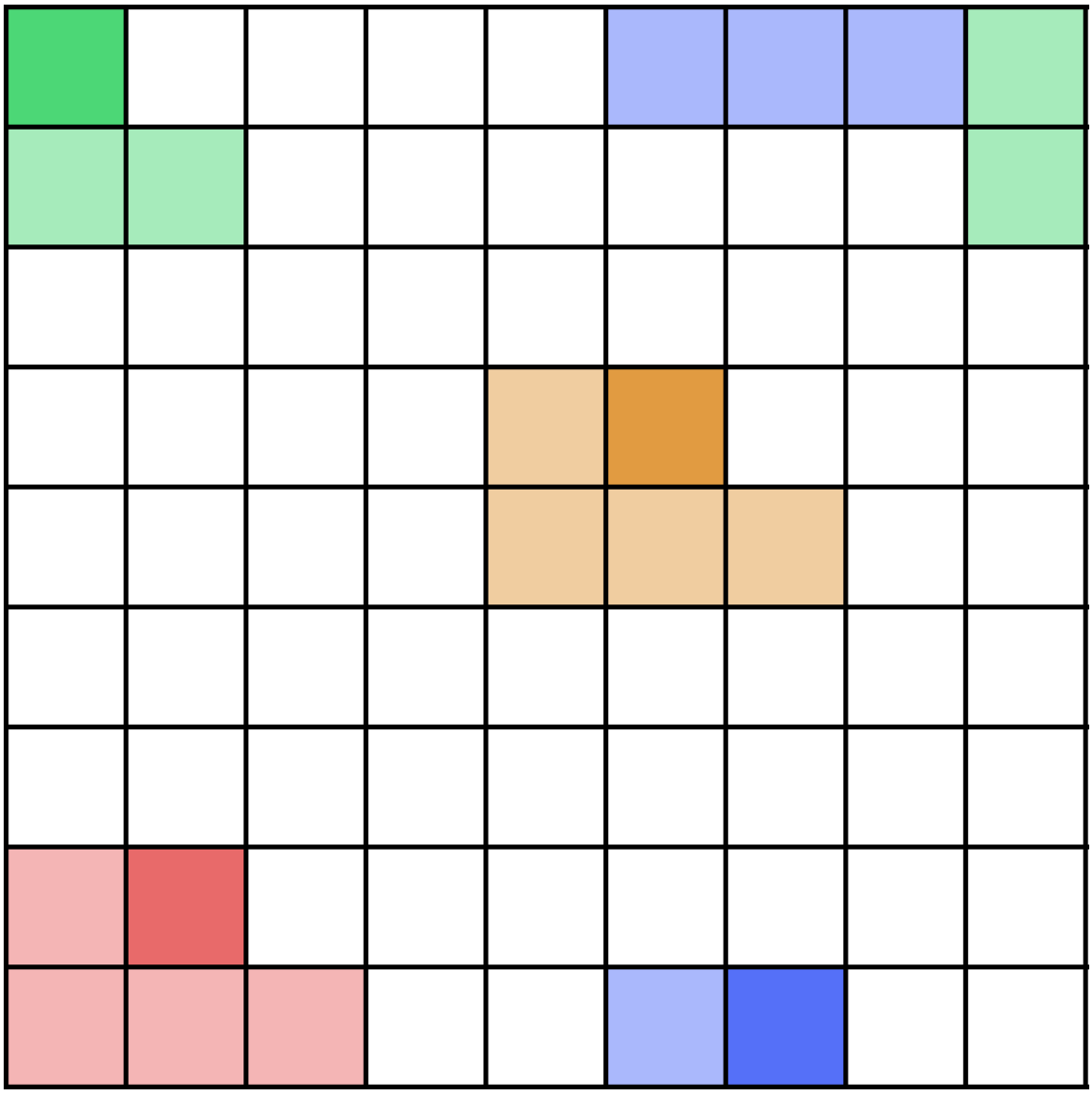
After all the neighbors have been found, the index lists are sorted in ascending order first by list 1, then by list 2, so the final sorted lists might look like this:

list 1	0	0	0	1	1	1	1	...	N - i	N - i	...
list 2	1	2	6	2	4	7	N	...	N - i + 1	N - i + 3	...

At first it might seem that the procedure explained above misses some possible comparisons, as the bubbles in each cell are compared to the bubbles in only half of the neighboring cells. This initial worry is easily dispelled, however, once one checks which cells e.g. the northern cell of c_x compares its bubbles to.

This concludes the inspection of the program's structure. Next, it is time to discuss the simulations performed with this model and the results and insights gained from the simulations.

Figure 3.7: The simulation box divided to equisized cells. Each color represents the neighborhood of cells to search for neighboring bubbles, where the cell "responsible" for the search is colored with a darker color. The sides of the grid have periodic boundary conditions in this example.



4. Simulation results

In this chapter, I will first show how changes in the gas fraction affect the behaviour of the foam. We will see typical scaling behaviour and indications of avalanche-like restructuring events, both heavily dependent on the gas fraction. These simulations also hint at some novel phenomena, which only arise with a sufficiently large system. The rest of this chapter is devoted to the study these new modes of behaviour.

4.1 Coarsening of two-dimensional foam

The time evolution of the average radius of bubbles in a froth is expected to follow a power law of the form

$$\langle R \rangle \quad \propto \quad \tau^{\alpha}, \tag{4.1}$$

where the exponent α depends on the liquid content of the foam and the shape of the individual bubbles [1]. Theory predicts that the exponent $\alpha = \frac{1}{3}$ for wet foams and $\alpha = \frac{1}{2}$ for dry foams [1]. To study the effect of varying gas fraction on the evolution, a batch of simulations was run with different gas fractions, i.e. different values of ϕ, with all the other parameters equal[†]. ϕ was varied between $0.80 - 1.00$ with a step size of 0.01. The complete list of input parameters for these simulations is given in listing B.1.

The figure 4.1 shows the evolution of average radius normalized with the average input radius, as the function of scaled time (see section 2.3.2 for a reminder of the definition of scaled time). From the figure it is visible how the exponent in the scaling law changes smoothly from 1/2 to 1/3 as the gas fraction decreases. In other words, the simulation captures the predicted change in behaviour of a self-similar foam as the amount of liquid is varied.

For an example of how the simulation captures the mechanical behaviour of foams, we should look at figures 4.2 - 4.5. These figures show the time evolution of three different variables for a particular value of ϕ: the total energy stored in bubbles E (in internal units), normalized change of energy ΔE and the normalized variance of

[†]Except of course for the random seed

Figure 4.1: Average radius as a function of scaled time. The two eye guides show that the exponent α varies between 0.33 and 0.5, depending on the gas fraction.

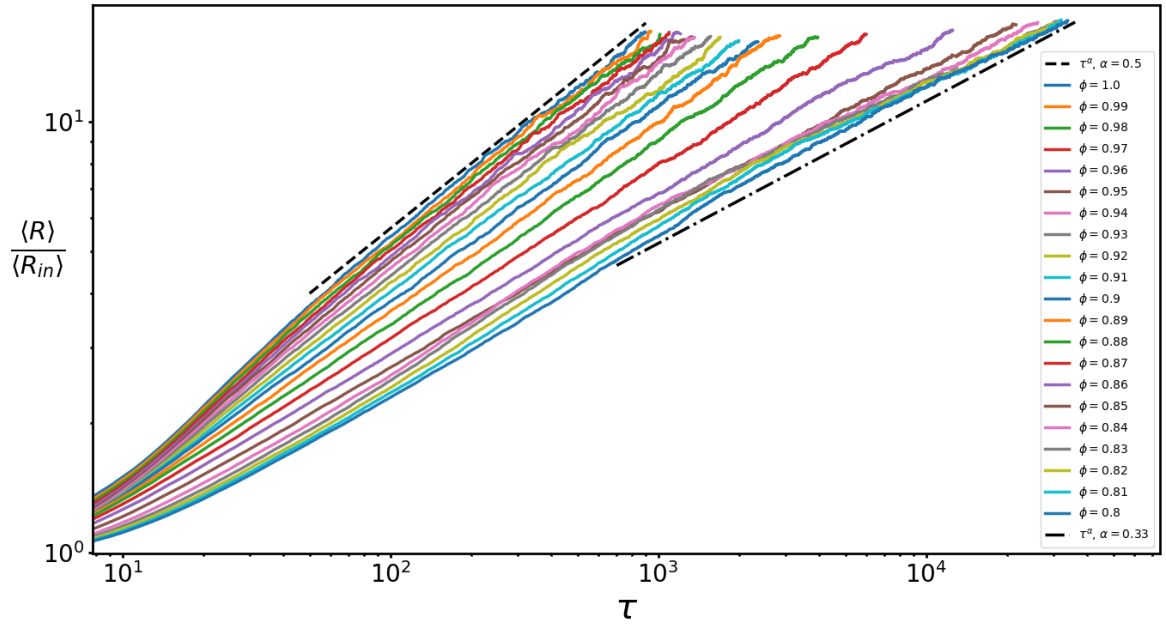


Figure 4.2: Total energy stored in bubbles, change of that energy and variance of velocity with $\phi = 1.00$. See text for discussion of scales.

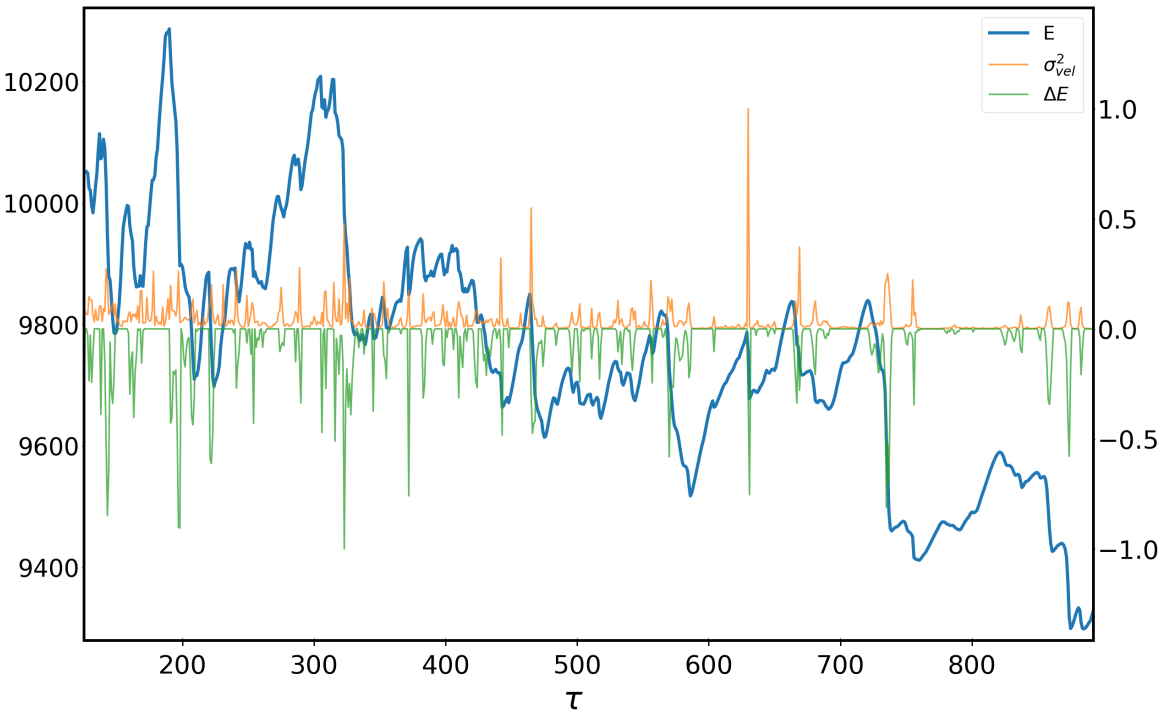


Figure 4.3: Same as figure 4.2, but for $\phi = 0.95$.

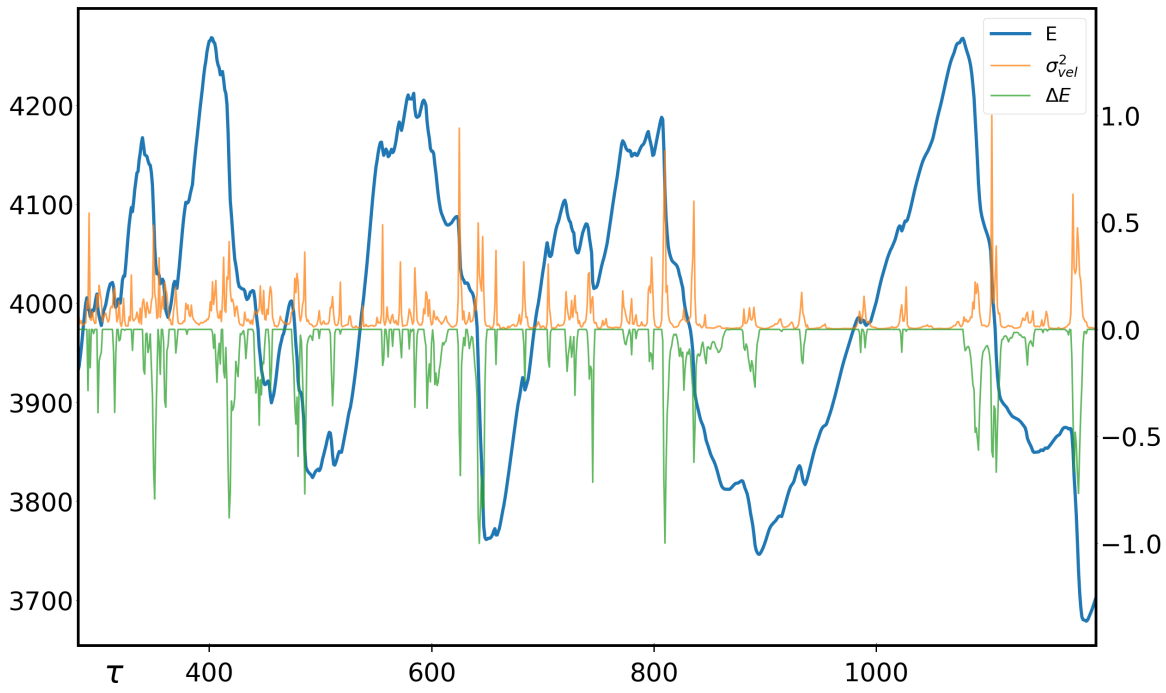


Figure 4.4: Same as figure 4.2, but for $\phi = 0.90$.

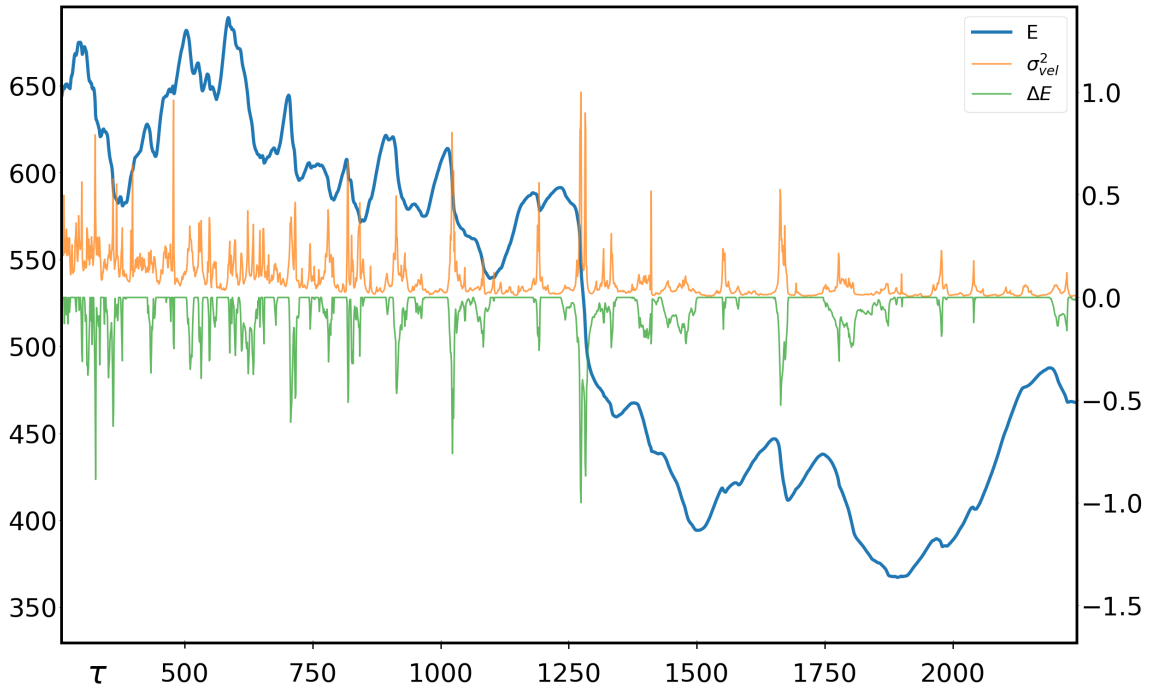
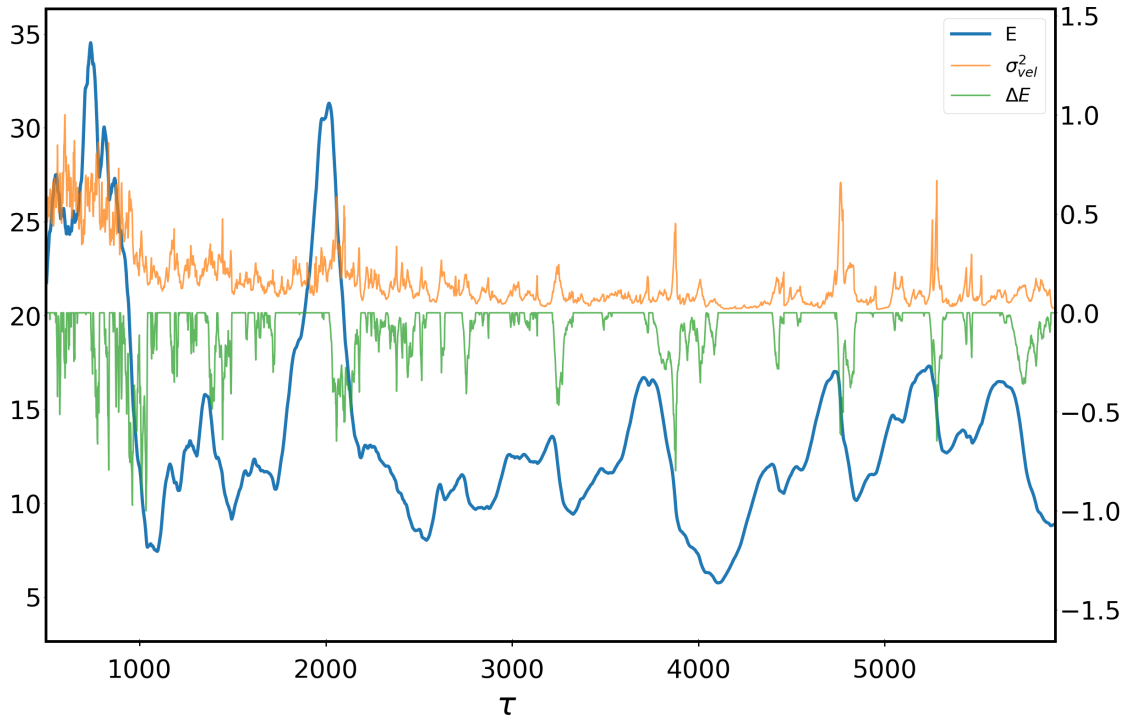


Figure 4.5: Same as figure 4.2, but for $\phi = 0.87$.

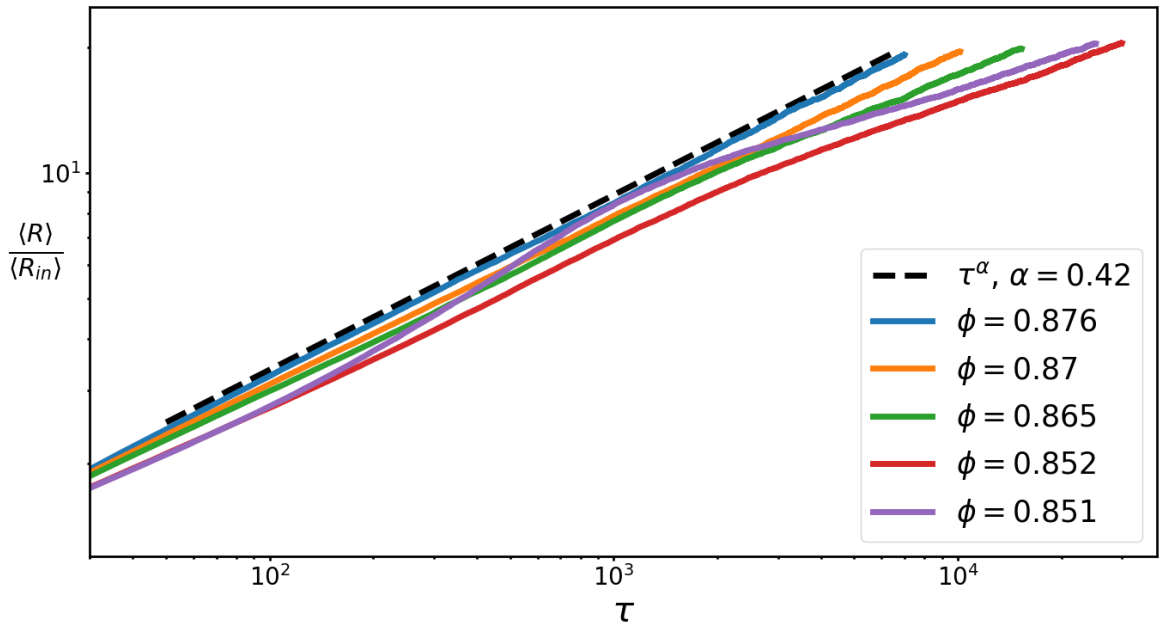


velocity σ^2_{vel}. The latter two values have been normalized so that the maximum absolute value is 1.0. In addition to being normalized, the positive values of the change in energy have been clipped to zero to better illustrate the symmetry between the two normalized values and to make the plots clearer.

A more detailed look at the figures reveals that the spikes coincide with abrupt releases of energy. This is hardly surprising in the case of ΔE, but the additional spikiness of σ^2_{vel} alludes to the existence of high-velocity bubbles. Taken as a whole, this behaviour indicates that the reconfiguration of the foam happens in short bursts or avalanches. Note that the spikes of σ^2_{vel} alone do not necessarily indicate an avalanche. Sometimes these high variances are caused by individual bubbles. One good example of this can be seen in figure 4.4 around $\tau = 500$, where there is a very tall spike in variance but only a tiny dip in energy.

Another thing to notice from the figures 4.2 - 4.5, is that the spikes get wider as the gas fraction decreases. The lower the gas fraction, the wider the spikes and the slower the changes in energy. In other words, the lower the gas fraction, the less these events behave like avalanches. Already around $\phi = 0.87$ the movement of bubbles in the foam starts to approach the smooth, slow and relaxed behaviour of lower gas fractions where sudden movements are rare if not nonexistent. As the jamming point of the foam varies somewhere close to 0.84 (see [26] and the references therein), approaching that limit means more and more free bubbles not in contact with other bubbles start

Figure 4.6: Average radius as a function of scaled time. Notice the large bend in the curve for $\phi = 0.851$



to appear. There is less energy stored in the foam and the forces between bubbles are smaller as the bubbles are less deformed.

As was alluded in the introduction of this chapter, some new and interesting things seem to be lying in wait beneath the surface. Returning to figure 4.1 and looking at the behaviour of the lowest gas fractions, it is seen, that the scaling of the foam with $\phi \leqslant 0.85$ seems to differ slightly from that of higher gas fractions. Instead of the very consistent straight lines the highest gas fractions seem to exhibit, there are slight "parabolic" bends in the curves. In other words, the steepness of the curves first increases, then decreases, i.e. the exponent α is not constant throughout the simulation. What can increase the scaling exponent? Why does it seem to only happen at lower gas fractions? Is there an underlying phenomenon beneath these effects, and if yes, what is it? These questions and more are investigated in the next section.

4.2 Phase separation in low gas fraction foams

To further investigate the anomalies in the evolution of the lower gas fractions, more simulations were done with a larger system size of 1×10^6 bubbles at the start of the simulation. ϕ was varied between 0.840 - 0.876 with a step size of 0.001. Figure 4.6 shows the scaling of a few of these to highlight the anomalous behaviour of some of the systems. The full list of input parameters used are listed in B.2*.

*Note that each simulation also used a unique random seed

The same "bending" of the scaling curve as seen in figure 4.1 is more clearly visible in figure 4.6 for $\phi = 0.851$ and less pronounced for $\phi = 0.865$. As seen from the curves for the rest of the gas fractions, there is no simple relationship between the increase in steepness of the scaling curve and the value of ϕ. To get a clearer picture of what is happening with the foam we can take a look at the actual state of the bubbles. Figure 4.7 shows an image of the foam at $\tau = 300$ for $\phi = 0.851$, $\phi = 0.852$, $\phi = 0.865$ and $\phi = 0.876$, from left to right, top to bottom. The images were rendered with ParaView from the generated snapshot data.

Apart from the obvious observation that there are large gaps between the bubbles, we can immediately notice two crucial details from these images. Firstly, there are no gaps visible at all on the snapshot of $\phi = 0.876$ and secondly, there is one large gap on the snapshot of $\phi = 0.851$ (due to periodic boundary conditions) but multiple small ones on the snapshot of $\phi = 0.852$. The first observation seems to be related to the fact that the "erratic" behaviour was noticed only when the gas fraction was sufficiently small, i.e. that foams with a high enough gas fraction do not exhibit a separation of liquid and gas phases. The second observation combined with the curves in figure 4.6 seems to indicate that the size of the gaps has a bigger influence on the evolution of foams than the mere existence of these gaps.

4.2.1 Effect of phase separation on foam evolution

So how is the phase separation related to the scaling exponent? Since the total gas fraction stays constant throughout the simulation, a lower gas fraction in one place must mean a higher gas fraction somewhere else. The more pronounced the phase separation is, the drier the remaining foam must be. In other words, larger areas of low energy bubbles means tighter packing of bubbles in the remaining areas. This effect is shown in figures 4.8 and 4.9. The figures show the distributions of absolute deviations of local gas fractions from the global value at different times. The simulation box was divided into a 10×10 grid and the local gas fraction was calculated in each. The difference starts to be very stark after $\tau = 60$.

Since the gas exchange mediated by the liquid phase has a significantly smaller contribution relative to the gas exchange through thin films*, the global gas exchange is dominated by the tightly packed areas. The combined effect is that the foam evolves as would a foam of a higher gas fraction, i.e. the average radius increases much faster than would normally be the case.

Eventually, the scaling exponent decreases towards the expected value for the global gas fraction. As the gas phase becomes drier and drier, the bubbles get packed

*See equation 2.29 and note that κ was 0.1 for these simulations

Figure 4.7: Snapshots of bubbles at $\tau = 300$ for $\phi = 0.851$, $\phi = 0.852$, $\phi = 0.865$ and $\phi = 0.876$, from left to right, top to bottom. The colors scale with the radius of bubbles.

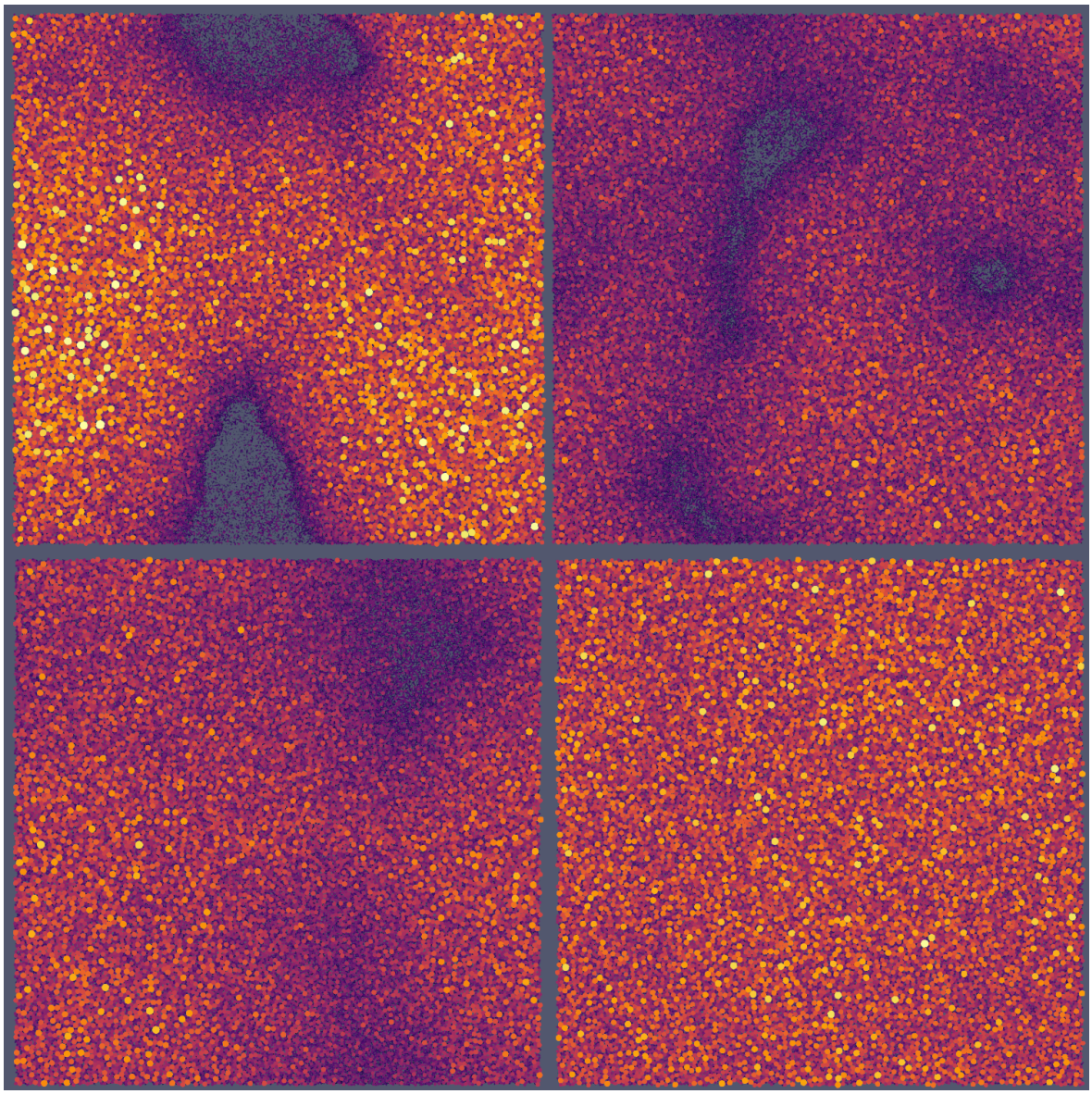


Figure 4.8: Distribution of local gas fraction at different time steps for different gas fractions.

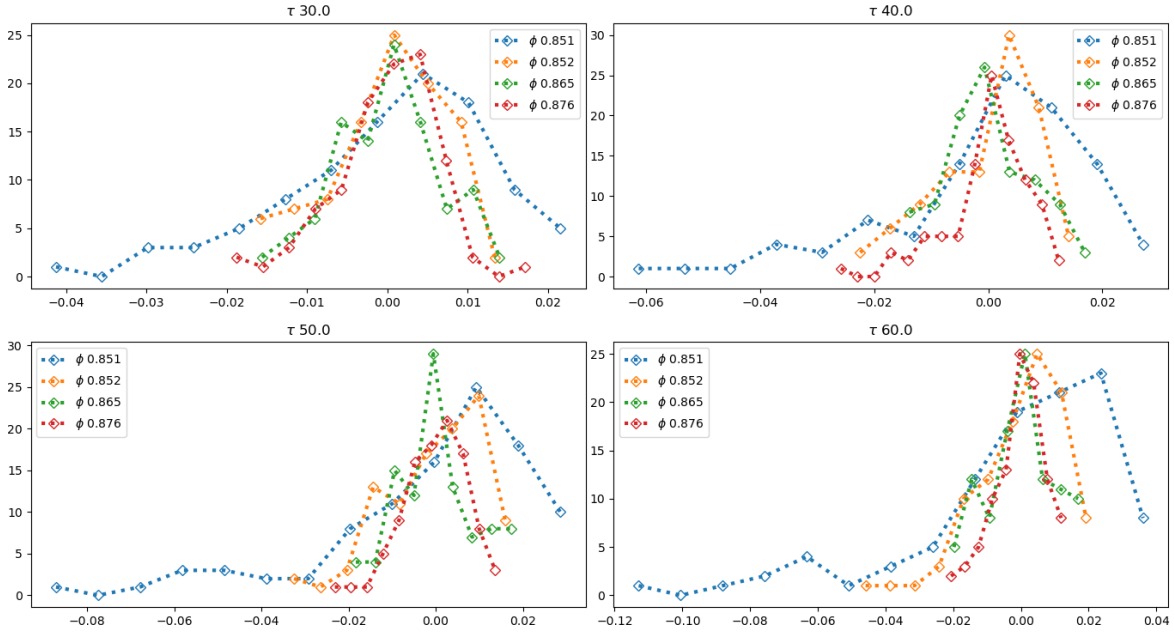


Figure 4.9: Distribution of local gas fraction at different time steps for different gas fractions.

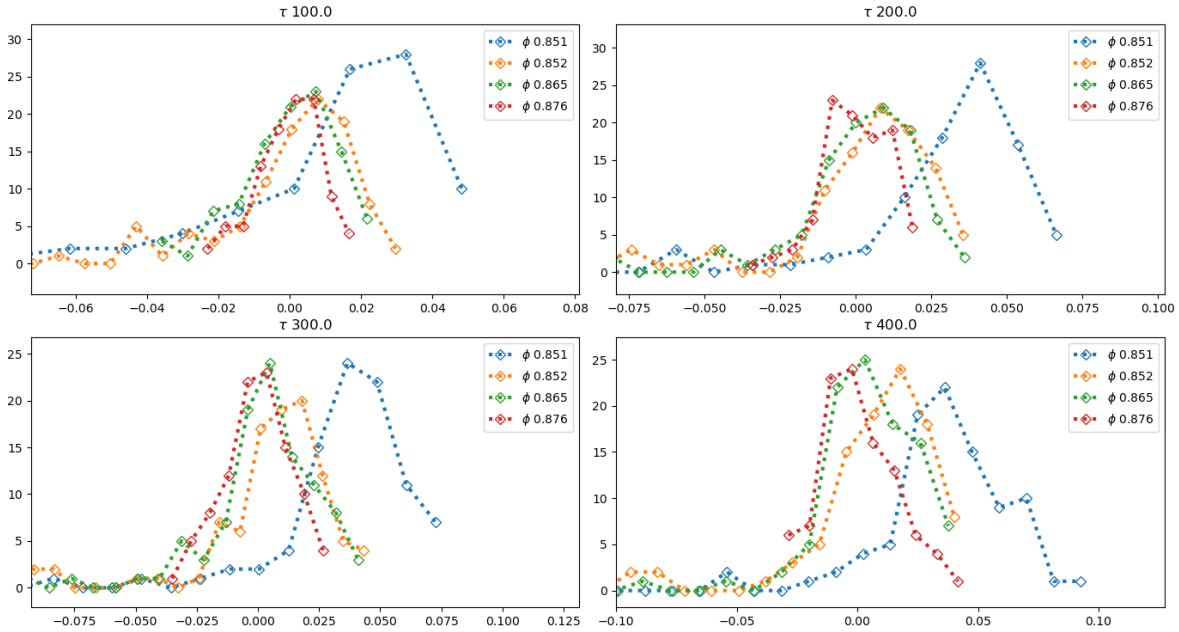
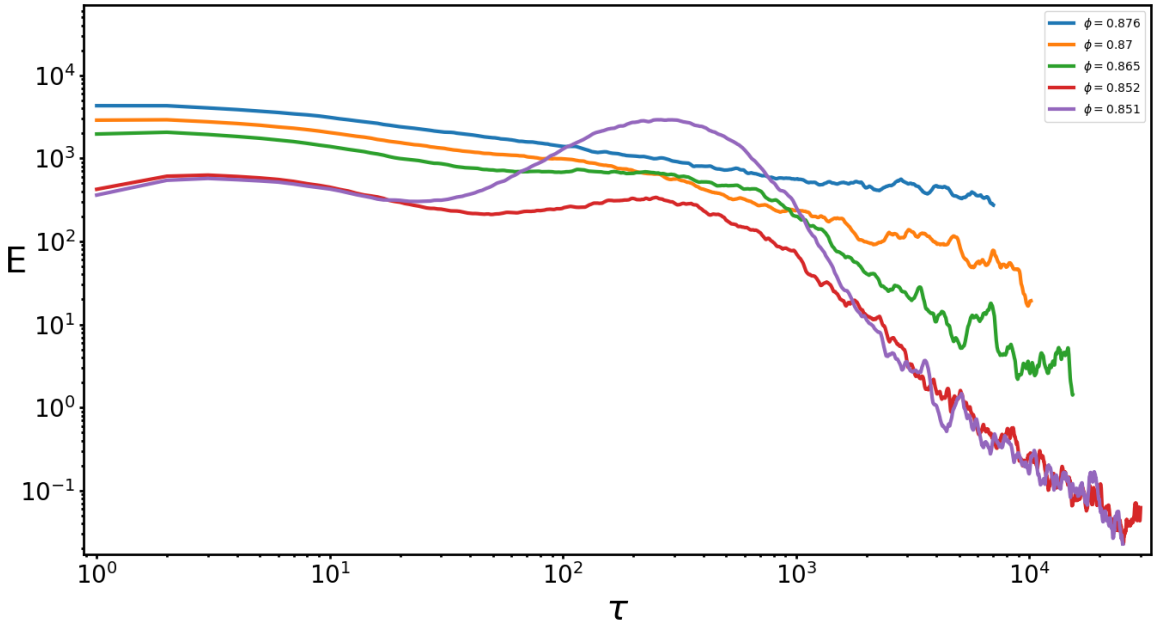


Figure 4.10: Total energy stored in bubbles in internal units.



tighter and tighter. This deforms the spherical shape and increases the tension between the bubbles and the energy stored in the foam. This, in turn, forces the foam to expand towards the empty regions, releasing energy and mixing the two phases slowly back together. As the foam homogenizes, the coarsening starts to resemble the expected behaviour for this gas fraction.

Figure 4.10 shows this from the energy perspective. The phase separation increases the energy stored in the foam, which is slowly released as the foam expands. The more pronounced the separation, the more energy is stored in the foam. A beautiful visualization of the expansion of the foam can be seen in figure 4.11. The colors show the distance that a bubble has travelled since the last time step. The bubbles are travelling towards the empty areas in clearly formed lines. Figure 4.12 shows the same for $\phi = 0.876$, where there is no structure and the motion of the bubbles appears random.

4.2.2 The origin of phase separation

So why does the phase separation happen in the first place? It seems to be related to the distribution of energy in the foam. When energy is distributed evenly across the foam, bubbles are tightly packed everywhere and the configuration of the foam is similar across the foam. The absolute rate of gas exchange does not vary much between different regions of the foam for a bubble of radius R.

When energy is distributed unevenly across the foam, there are regions where bubbles are packed more tightly and conversely regions where they are packed more

Figure 4.11: Distance the bubbles have moved since the last time step for $\phi = 0.851$ at $\tau = 100$.

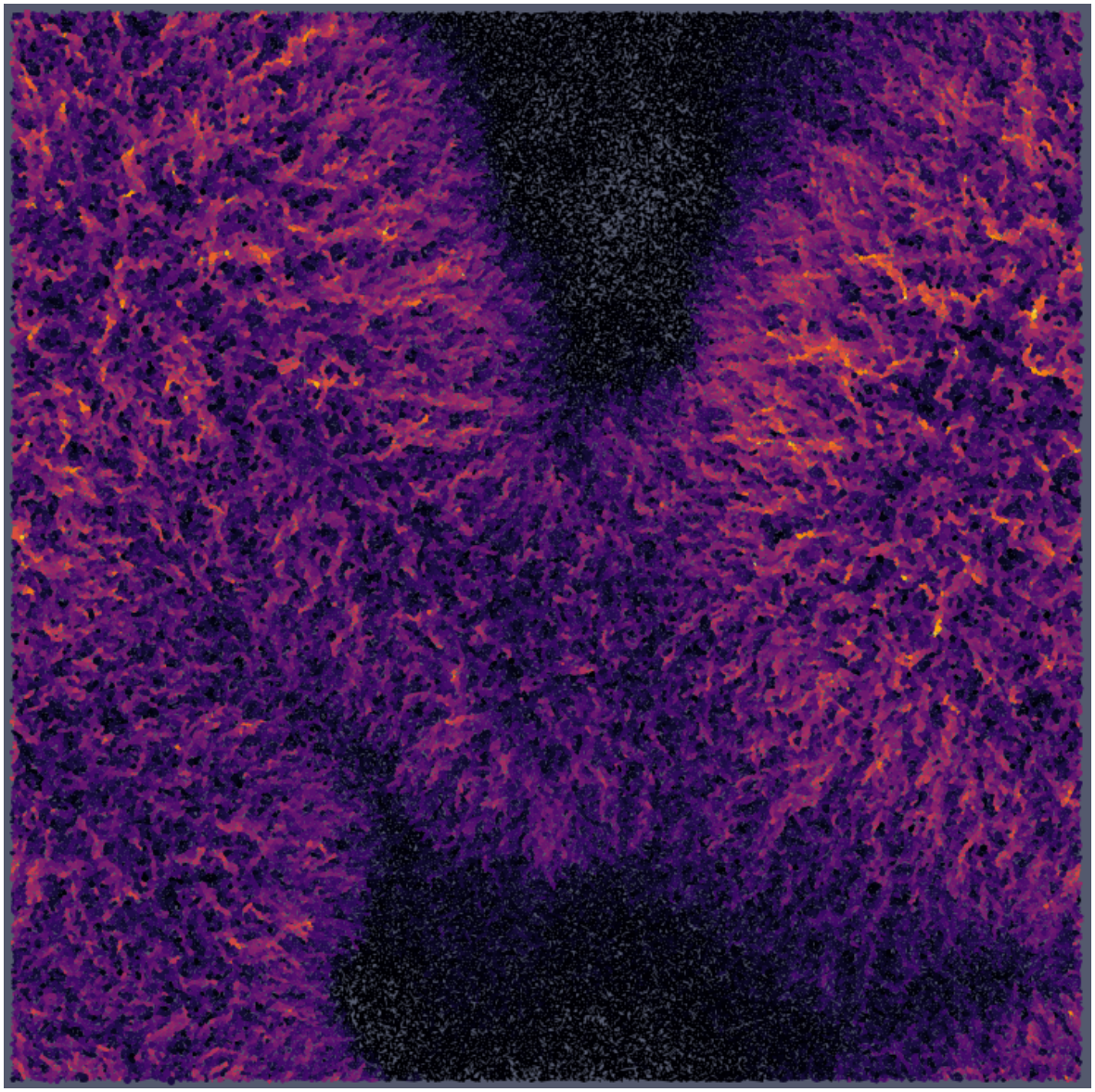


Figure 4.12: Distance the bubbles have moved since the last time step for $\phi = 0.876$ at $\tau = 100$.

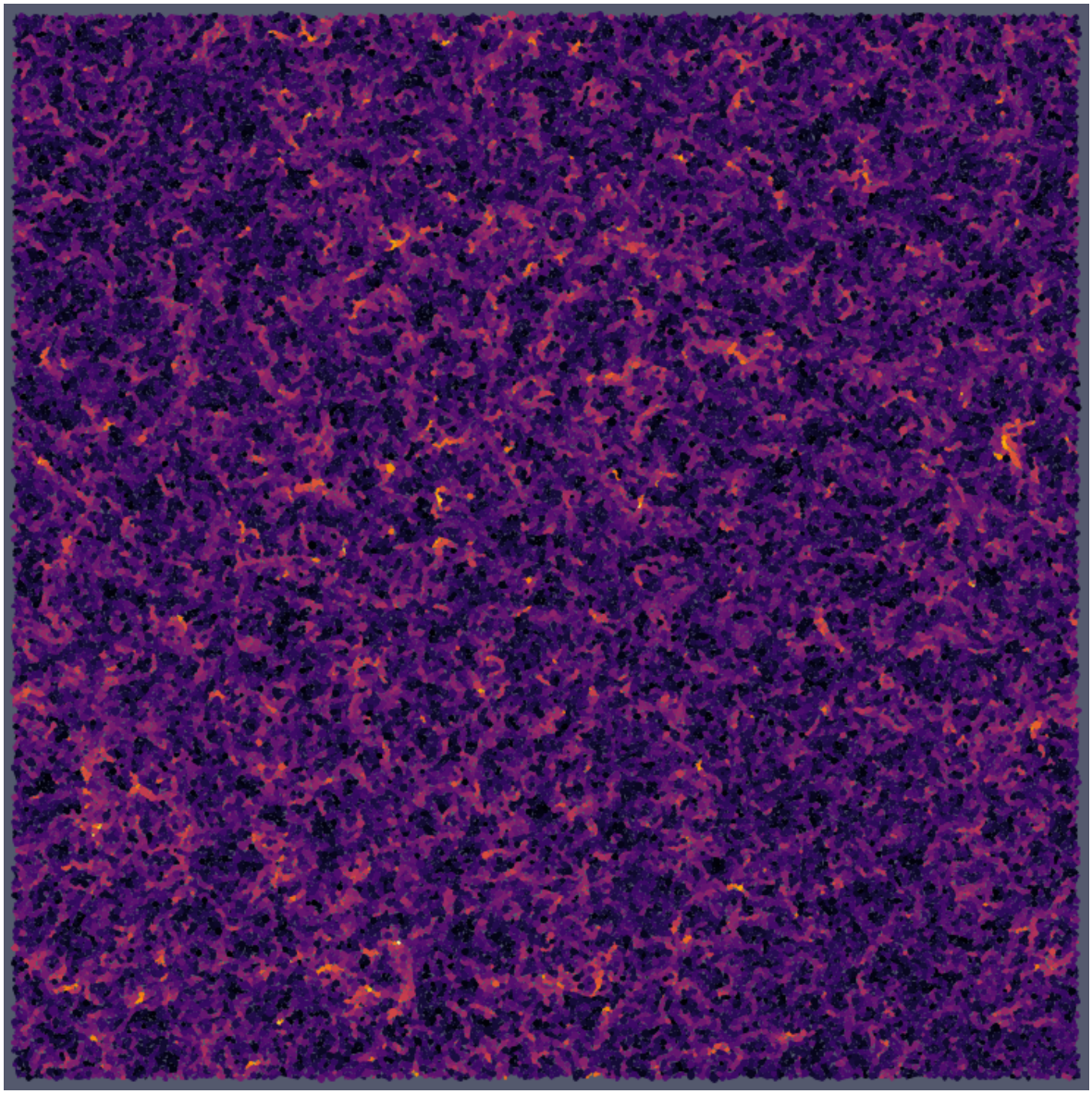
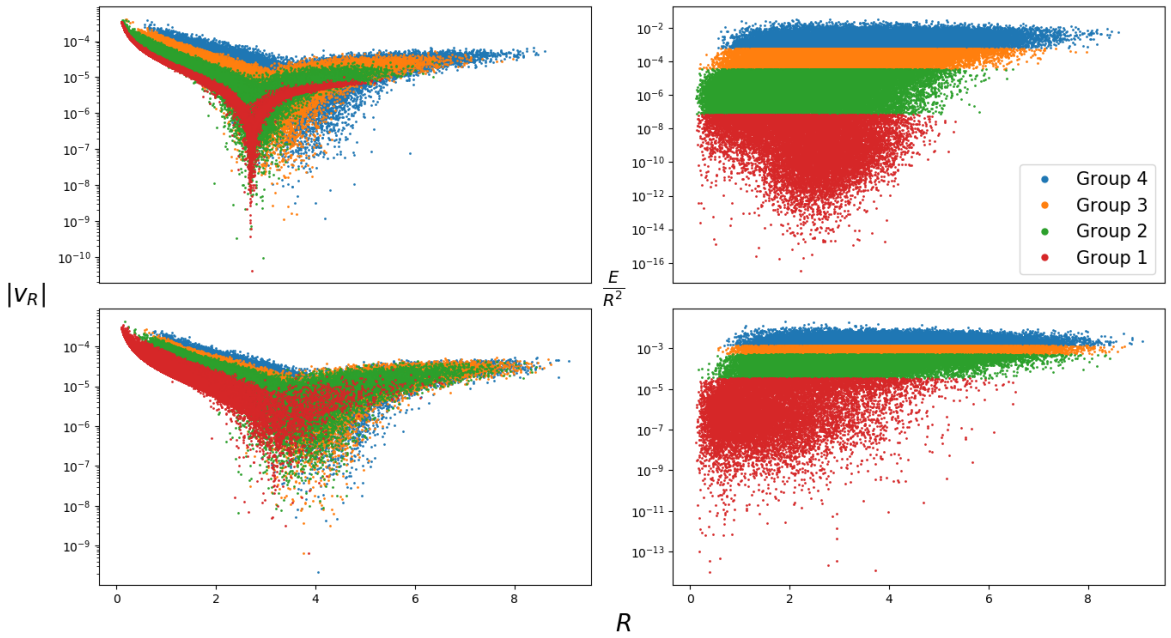


Figure 4.13: Absolute rate of change of radius as a function of radius for four different groups. See text for explanation of groups. The upper plots are of $\phi = 0.851$ and the lower of $\phi = 0.876$ at $\tau = 100$.



loosely. Areas of the foam can get locally jammed, while others consist of almost neighborless bubbles. This means the rate of gas exchange for a bubble of constant size varies across the foam, because the bubbles in loose regions have fewer neighbors and the areas of contact are smaller with the few neighbors they have*. Thus the absolute rate of change of the radius of a bubble is smaller on average in loosely packed areas, compared to bubbles of similar size in the locally jammed areas.

This effect is visible in figure 4.13. The plots on the left show the absolute rate of gas exchange as a function of radius for four different groups of bubbles. The plots on the right show the energy density as a function of radius. The bubbles were divided into four groups based on their energy density (that is why there are clear boundaries on the right plots). Group 1 contains the lowest 25%, group 2 the next lowest and so on. As the figure shows, there is greater variance across the different groups, when the energy distribution of the foam is uneven (the upper plots, for $\phi = 0.851$).

In the upper left figure, group 1 (the least compressed bubbles) is distributed narrowly, centered on a particular value of the rate of gas exchange. At roughly $R = 3$ the rate of gas exchange dips towards zero. As these bubbles have zero or few neighbors, the dip in the rate means their radii are close to the value of ρ in equation 2.29. As the energy density increases, the bubbles are distributed across a wider range of rates of gas exchange. Still, there are clear differences between bubbles of group 2 compared

*See equation 2.29

to the most compressed bubbles of groups 3 and 4. For example, the maximum rates of gas exchange are significantly higher among the more compressed bubbles.

Conversely on the lower left figure, which shows the behaviour in a foam of a more even energy distribution ($\phi = 0.876$), the differences between the groups are much less pronounced. There are hardly any differences between groups 2, 3 and 4 and the behaviour of even the least compressed bubbles is quite similar to the rest. These similarities stem from the fact that there is much less variance between the four groups overall, as comparing the ranges of energies the bubbles are distributed at in the plots on the right shows.

Further comparing the plots on the right we can make some more observations. Firstly, the upper 75% of bubbles are spread out on a much less narrow band of energy densities in the lower plot, as already pointed out. Secondly, in the upper plot, the largest bubbles are consistently among the most compressed, and the maximum size of the group decreases as the energy density decreases. This is not as clearly so in the lower plot. Lastly, the lowest energy densities are at very different radii between the two foams. For the foam with a more even energy distribution, the least compressed bubbles are the smallest ones. This is because they find their places in the gaps between the large bubbles and can have essentially zero compression even in densely packed areas. For the foam with separate phases, the least compressed bubbles are packed around the radius at which the rate of gas exchange approaches zero. In this case, most of the least compressed bubbles are in the areas of low density. Their gas exchange is dominated by the liquid intermediated term, as they have very few if not zero neighbors, and the areas of contact with those few neighbors are small.

The reason most of the free bubbles are bunched up close to the radius ρ, which is the radius for the average bubble the gas content of the liquid is modelled as, is related to the way bubbles of different sizes behave in the loose regions. For bubbles that are smaller than the average bubble, the rate at which they shrink is faster, the smaller they are. In other words, the less volume they have, the faster they lose their remaining volume, and thus the speed at which they shrink accelerates. Thus the radius ρ works as a cutoff radius in the loose regions: once a bubble is below that radius, it will shrink at an ever accelerating radius until its eventual demise. The bubbles that are larger than the cutoff radius increase in size, albeit very slowly. As the rest of the foam evolves and the average size of all the bubbles keeps increasing, so must the cutoff radius ρ increase. Since the average bubble increases faster than the slowly increasing bubbles with $R > \rho$ in loose regions, at some point the average bubble surpasses these bubbles and they end up shrinking as well. So the radius of the bubbles in the loose regions is centered on the value of ρ, because a region of very slow evolution forms around the radii close to it, while it keeps ever increasing in size, thus ensuring that the larger

bubbles with slower rates of change do not "run away" from it.

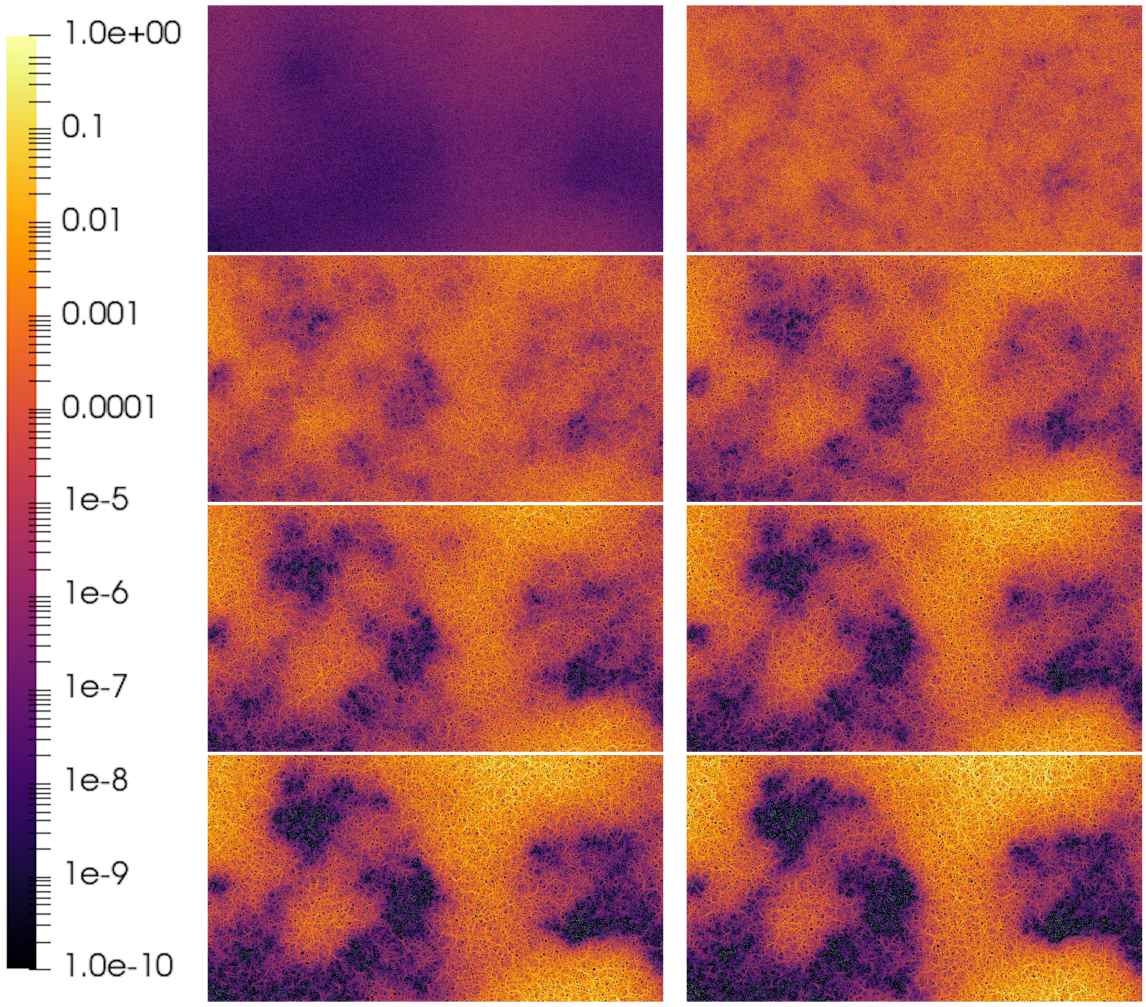
This small imbalance in the rate of gas exchange between different regions of the foam can lead to larger imbalances fairly quickly, because a positive feedback loop is formed. Bubbles in loose regions exchange gas slower than bubbles in tight regions. This leads to the large bubbles in jammed regions to increase in size faster than bubbles of equal radius in the looser regions. Thus the fastest growing and largest bubbles will be found in the tightly packed regions while more and more bubbles in the looser regions become smaller than the average, because the bubbles there either shrink, stay constant in size or grow slower than bubbles of similar size in other regions, while the average size keeps growing. As the neighbors of larger than average bubbles slowly shrink around them in the loose regions, the large bubbles end up being isolated and surrounded by the liquid phase, thus all but stopping their growth and dooming them to the inevitable fate of being surpassed in size by the average bubble and thus forcing them to shrink. As the last bubble in the originally slightly less tightly packed area disappears, the phase separation is complete and the now-empty region awaits the proverbial charge of the surrounding foam.

Figure 4.14 shows snapshots of energy distribution for a foam with $\phi = 0.840$. The colors are on a logarithmic scale. The first image shows the slightly uneven distribution at the start of the simulation, which later leads to the formation of empty regions and thus the phase separation in the foam.

4.2.3 Distribution of energy and the gas fraction of the foam

If an uneven energy distribution lies at the heart of the phase separation, what can lead to that uneven distribution in the first place? When the gas fraction is sufficiently high (above the jamming point "J" [27]), the equilibration process drives the foam towards a state where there is minimal change in energy between two relaxation rounds. At the end of the equilibration, the entire foam is in a relatively stable state, where the forces between neighboring bubbles balance each other out, where the smallest bubbles ("rattlers") are in the gaps between larger bubbles and where networks of long, entire foam spanning force chains are formed, much like in other systems consisting of deformable granular material [28]. With the entire foam being jammed and connected by force chains, local variations can have a global effect. To visualize this, imagine a situation, where the entire foam is in this kind of stable state, and the size of one of the rattlers is increased. The bubbles around the ex-rattler now feel a significant force and exert this force on their neighbors. Rattlers by definition are not part of the force chains, and increasing one to a significant size would exert a force that is not parallel to the force in the chain. As anyone who has played with beads or marbles knows, any

Figure 4.14: Energy stored in bubbles on a logarithmic scale for $\phi = 0.840$ at $\tau = 0$, 10, 20, 30, 40, 50, 60 and 70 left to right, top to bottom.



perpendicular force to a chain of spheres under tension very easily buckles the entire chain. When these force chains buckle, the foam can get momentarily unjammed, which is seen as the release of potential energy, before jamming again to a new configuration. These avalanche-like reconfiguration events were briefly discussed in section 4.1. In summary, a high gas fraction foam self-organizes to a state with long force chains consisting mostly of large bubbles with the smallest bubbles having been driven to the gaps between larger bubbles. The foam surrounding the force chains works to prevent the chains from buckling, which, when it happens, leads to a brief unjamming and reconfiguration of the foam to a new jammed state [29] [30] [31].

When the gas fraction is sufficiently low (below the jamming point), equilibrating the foam does not lead to a foam that is connected through and through. In this kind of foam, some bubbles might not even be in contact with their neighbors, and consequently, local changes have little to none global effect, because local disruptions at one location have no way of propagating to other areas. Additionally, if the gas fraction of the foam would be increased by first fixing the positions of the bubbles and then shrinking the entire simulation box, the state of the foam would most certainly be very far from a stable state for this new gas fraction, because the bubbles have not been forced to positions that minimize the energy of the foam when the bubbles are in tight contact with their neighbors. The smallest bubbles have not been forced to the positions of rattlers and they can even form small clusters. The consequence of these matters is that the size and energy distributions of the foam can be uneven already at the start of the simulation for the low gas fraction foam because the loose packing of the bubbles affords the foam very nonoptimal configurations. When the simulation starts and the coarsening is turned on, areas with large bubbles can suffer localized jamming, whereas areas of small bubbles may start forming very loose areas and the phase separation starts to take place.

A more detailed study of the effect of bubble distribution for low gas fraction foams could help to determine the veracity of this idea. One way to study it was already hinted at: equilibrating the foam as a foam of higher gas fraction, then decreasing the gas fraction by keeping the normalized positions of the bubbles constant while increasing the size of the simulation box. This way the bubbles would be distributed similarly to a foam of higher gas fraction and the effects discussed at the end of section 4.2.1 might not have a chance to manifest themselves.

4.2.4 Distribution of energy and the size of the system

Why does the phenomenon manifest itself only with sufficiently large system sizes? Taking a look at the figures 4.6, 4.7 and 4.10 can give us an indication of the reason.

Earlier we observed that the size of the low energy region seems to have a big role in the subsequent evolution of the foam. The evolution of the foam with $\phi = 0.852$ was very different from that of $\phi = 0.851$, and the most notable difference between them was the size and number of the low energy regions. The energy of the foam with many small low energy regions rose only slightly before decreasing back to its expected levels, and this did not have as drastic an effect on the scaling of the average radius as with the foam with one large low energy region. This is because the expanding foam can reclaim the empty regions much faster when the regions are small; there is less ground to cover so it happens faster. Because the foam homogenizes earlier and faster, the phase separation has less of an effect on the evolution of the foam as a whole.

So clearly the size of the low energy region plays a significant role in the phase separation. But how is the size of the low energy region related to the size of the system? As the system size is increased, the ratio of the sizes of the average bubble and the simulation box decreases. Due to geometrical reasons, filling the same space with deformable spheres deforms the spheres a lot less, if the spheres are small compared to the encompassing space. This means that larger systems can afford more and larger loosely packed areas because the "negative space" divided among the rest of the system does not deform the spheres (i.e. increase the energy stored in the bubbles) as much as it would in a system with larger ratio (that is, smaller size of simulation box relative to the average size of the bubbles). As the ratio increases, the ability of the system to support low energy regions decreases, until at some point it is no longer possible at all.

In summary, a low gas fraction means a greater possibility for uneven distribution of energy across the foam, because the foam can support very nonoptimal configurations of bubbles which may lead to localized jamming. A large system with uneven distribution of energy means a greater possibility for large regions of low energy, which in turn can lead to the temporary separation of gas and liquid phases. Separation of phases causes the foam to evolve for a while as would a foam of greater gas fraction, before reverting to its normal behaviour once the phases have mixed back together.

5. Conclusions and outlook

In this thesis, I have presented a GPU implementation of a wet foam model programmed using Nvidia's CUDA and modern C++. In chapter 2 I presented the theoretical background supporting the implementation of the simulation code. Here we learned about two different models for foams, firstly of the theory by Lifshitz, Slyozov and Wagner modelling the behaviour of wet foams, where bubbles are roughly spherical in shape and not in contact with each other [3], [4]. The second theory we learned about was by von Neumann describing the behaviour of very dry foams, where the bubbles are polygonal or polyhedral in shape, depending on the dimensionality of the foam, and tightly in contact with each other [5]. We learned that these two theories give different results for the coarsening of the foam, which was not entirely surprising, given the fact that the structures of the foams are very different.

After these two theories, I also introduced several different ways the behaviour of foams have been simulated in the past. Some of these simulations were concentrated on dry two-dimensional foams, some were extensions of earlier simulation schemas which originally had nothing to do with the simulation of foams, and some were related to the simulation of wet foams. We had a detailed look at the wet foam model proposed originally by Durian, of which this simulation is an extension [8]. The chapter was concluded with the two important equations on which this simulation code leans on: the equation of motion for the bubbles as well as the equation for gas exchange (equations 2.15, 2.29, respectively).

After the theoretical part, we moved on to chapter 3, where we dove right into the implementation details and structure of the code. We started the chapter with a discussion on the differences between GPU and CPU programming and how the memory layout of the program has an impact on the performance of the code. After this technical start, we went through the structure of the simulation, explaining each part of the program as we proceeded. We saw how the simulation starts with scaling and stabilization phases following the initial setup of the simulation and its parameters. After these mandatory setup steps, the simulation then proceeded to the simulation loop, where integration, neighbor search and bubble removal took place.

In the last chapter, we looked at the behaviour of two-dimensional foams as simu-

lated with the presented code. At first, we saw the usual behaviour for coarsening and mechanical responses of the foam with different gas fractions. Then we investigated a novel phenomenon that arose at low gas fractions with large system size and attempted to explain its behaviour and origin. We learned that the simulated foam exhibited a temporary separation of the gas and liquid phases due to the uneven distribution of stored energy in the foam. As time proceeded, the phases mixed back together and the evolution of the foam returned to the expected patterns.

As is usually the case, studying something in detail raises more questions than answers them. Originally the plan for this thesis was to study the behaviour of many different systems with the presented code, including the differences between force and pressure-driven foam flows in two and three dimensions or the effect of varying the strength of the liquid intermediated gas exchange (i.e. the κ parameter). After some quick preliminary analysis, I did not find anything interesting when κ was modified while keeping ϕ constant. But the problem was that ϕ in those simulations was always above 0.87, i.e. sufficiently high that there was never going to be phase separation with those foams. Now that some of the reasons behind phase separation have been analyzed, it would be interesting to find out how different situations affect phase separation. Is it stronger or does it happen faster in some situations? How would it change with varying κ or with different original size distribution? Does an imposed flow make it more or less likely and how does it change the lifetime of the separation? Answering some of these questions, making statistical studies of these types of foams, using even larger system sizes and more deeply analyzing the possible effects of the assumptions inherent in the presented code are bubbles in the froth of foam physics that this thesis is a neighbor to.

Lastly, I will mention that my sincere hope is for the simulation code to be of some use to anyone who gains access to it, be it in the form of performing simulations and learning about the behaviour of foams or understanding something new about GPU programming or physical simulations in general. If nothing else, at least this code allowed me to learn a great deal about both of the aforementioned subjects, which I found very rewarding. May your simulations run quickly and without bugs.

Acknowledgements

I extend my gratitude to my thesis advisor Antti Puisto for his support and insightful comments while I was working on this thesis and for hiring me to work on the code at the Complex Systems and Materials group of Aalto University. I also want to thank the rest of the members of the said group for the nice work environment and the volleyball and "koira" games.

Thanking my parents Tarja and Pekka is also in order. They have always supported my endeavours and provided me with guidance (and patience when I've utterly ignored said guidance). They also let me spend a week at their house in the autumn of 2019 while I was finishing this thesis. It was very productive and raking the leaves from the yard offered a nice way to get my mind off of bubbles.

Appendix A. Program I/O

A.1 Input

The program handles input by reading in a JSON file and storing the arguments as member variables of a single struct called "SimulationInputs". This struct is passed as a reference to the functions that need these values. New inputs can be defined very easily, by just adding a new variable to both the input struct and the input json. The complete input json is shown in the listing A.1.

A.2 Output

The program outputs two kinds of data files. First type is a snapshot of the state of the simulation containing the position and radius of each bubble among other variables. The writing frequency of the snapshots can be controlled, so that more snapshots can be written to generate a smooth animation of the simulation, or all the snapshots can be skipped entirely to save space. The data contained in a snapshot can be used to render the bubbles with ParaView or similar data visualization program. Listing A.2 shows an example of a snapshot file.

The second kind of output is a file with average data collected throughout the simulation. The data is piped to a `C++ stringstream` at constant intervals and when the end condition of the simulation is encountered, the `stringstream` is written to a file. What data is gathered can be determined by adding or removing variables from the data stream. Listing A.3 shows an example of the collected data. The variables in this particular case are (from left to right): scaled time, average radius of bubbles relative to input average radius, maximum bubble radius relative to input average radius, number of bubbles left in the simulation, the average path travelled since the beginning of the simulation, the average squared distance from the starting position, and the change in total potential energy since the last time step.

LISTING A.1: Input parameters of the program

```
{
    "avgRad": 1.0,
    "avgRadExpl": "The average radius of the bubbles.",
    "boxRelDim": {
        "x": 1.0,
        "y": 1.0,
        "z": 1.0
    },
    "boxRelDimExpl": "The relative dimension of the box.",
    "errorTolerance": 1e-05,
    "errorToleranceExpl": "The error tolerance level for the
        integration.",
    "flowLbb": {
        "x": 0.0,
        "y": 0.475,
        "z": 0.0
    },
    "flowLbbExpl": "(Relative) Lower-back-bottom of flow area.",
    "flowTfr": {
        "x": 1.0,
        "y": 0.525,
        "z": 0.0
    },
    "flowTfrExpl": "(Relative) Top-front-right of flow area.",
    "flowVel": {
        "x": 0.01,
        "y": 0.0,
        "z": 0.0
    },
    "flowVelExpl": "Flow velocity inside flow area.",
    "kParameter": 0.001,
    "kParameterExpl": "A bundle of constants related to gas exchange.
        Should be small relative to AvgRad.",
    "kappa": 0.1,
    "kappaExpl": "Relative strength of liquid-intermediated diffusion
        versus diffusion through thin films.",
    "maxDeltaEnergy": 1e-05,
```

```
    "maxDeltaEnergyExpl": "Maximum allowed energy change when relaxing
       the bubble bath.",
    "minNumBubbles": 300,
    "minNumBubblesExpl": "The simulation runs as long as there are at
       least this many bubbles.",
    "muZero": 1.0,
    "muZeroExpl": "The viscosity of the fluid",
    "numBubblesIn": 100000.0,
    "numBubblesInExpl": "Total number of bubbles to simulate.",
    "numBubblesPerCell": 32,
    "numBubblesPerCellExpl": "How many bubbles per cell.",
    "numStepsToRelax": 10000.0,
    "numStepsToRelaxExpl": "How many steps to take per 'relaxation
       round' when equilibrating the bubble bath.",
    "phiTarget": 0.95,
    "phiTargetExpl": "The target volume fraction.",
    "rngSeed": 4,
    "rngSeedExpl": "The seed for the RNG.",
    "sigmaZero": 1.0,
    "sigmaZeroExpl": "The surface tension (spring constant) of the
       fluid.",
    "snapshotFrequency": 1.0,
    "snapshotFrequencyExpl": "How often are snapshots saved. Bigger
       number can be used to get smoother animations, 0.0 means no
       snapshots are saved."
    "stdDevRad": 0.21,
    "stdDevRadExpl": "The standard deviation of the bubble radii.",
    "timeStepIn": 0.001,
    "timeStepInExpl": "The initial time step with which to increment
       the simulation time.",
    "wallDragStrength": 0.1,
    "wallDragStrengthExpl": "How strong the drag from the wall is.
       Should be between 0.0 and 1.0"
}
```

LISTING A.2: Snapshot

```
x,y,z,r,vx,vy,vz,vtot,path,distance,energy,displacement,index
0.16882,0.637941,0,0.851427,1.8741e-06,5.05712e-07,0,1.94113e
    -06,0,0,0.021086,0.659901,0
1.80229,1.05602,0,0.927044,1.8659e-06,4.43203e-07,0,1.91782e
    -06,0,0,0.029335,0.804239,318
3.7718,0.809576,0,1.14987,1.89338e-06,3.62805e-07,0,1.92783e
    -06,0,0,0.0403189,1.11852,319
5.81326,1.27026,0,1.02223,1.89658e-06,1.86719e-07,0,1.90575e
    -06,0,0,0.0330418,0.856988,320
7.10223,0.564915,0,0.477919,2.00864e-06,2.16767e-07,0,2.0203e
    -06,0,0,0.00988725,0.57409,321
--- snip ---
```

LISTING A.3: Collected data

```
1 0.983026 2.11807 99545 0.436157 0.136383 0.0499424
2 0.987104 2.30372 95128 1.01239 0.432933 -0.0117717
3 1.01392 2.53429 87756 1.62255 0.769102 -0.0145149
4 1.05786 2.71979 79295 2.24353 1.13189 -0.0169361
5 1.10728 2.90749 71475 2.85378 1.50119 -0.00965786
6 1.16325 3.09452 64285 3.46826 1.89532 -0.00793891
--- snip ---
```

Appendix B. Simulation inputs

LISTING B.1: Inputs for varying volume fraction in 2D


```
{
    "avgRad": 1.0,
    "avgRadExpl": "The average radius of the bubbles.",
    "boxRelDim": {
        "x": 1.0,
        "y": 1.0,
        "z": 1.0
    },
    "boxRelDimExpl": "The relative dimension of the box.",
    "errorTolerance": 1e-05,
    "errorToleranceExpl": "The error tolerance level for the
        integration.",
    "flowLbb": {
        "x": 0.0,
        "y": 0.475,
        "z": 0.0
    },
    "flowLbbExpl": "(Relative) Lower-back-bottom of flow area.",
    "flowTfr": {
        "x": 1.0,
        "y": 0.525,
        "z": 0.0
    },
    "flowTfrExpl": "(Relative) Top-front-right of flow area.",
    "flowVel": {
        "x": 0.01,
        "y": 0.0,
        "z": 0.0
    },
    "flowVelExpl": "Flow velocity inside flow area.",
    "kParameter": 0.001,
    "kParameterExpl": "A bundle of constants related to gas exchange.
        Should be small relative to AvgRad.",
    "kappa": 0.1,
    "kappaExpl": "Relative strength of liquid-intermediated diffusion
        versus diffusion through thin films.",
    "maxDeltaEnergy": 1e-05,
```

```
    "maxDeltaEnergyExpl": "Maximum allowed energy change when relaxing
       the bubble bath.",
    "minNumBubbles": 300,
    "minNumBubblesExpl": "The simulation runs as long as there are at
       least this many bubbles.",
    "muZero": 1.0,
    "muZeroExpl": "The viscosity of the fluid",
    "numBubblesIn": 100000.0,
    "numBubblesInExpl": "Total number of bubbles to simulate.",
    "numBubblesPerCell": 32,
    "numBubblesPerCellExpl": "How many bubbles per cell.",
    "numStepsToRelax": 10000.0,
    "numStepsToRelaxExpl": "How many steps to take per 'relaxation
       round' when equilibrating the bubble bath.",
    "phiTarget": 0.95,
    "phiTargetExpl": "The target volume fraction.",
    "rngSeed": 1231234,
    "rngSeedExpl": "The seed for the RNG.",
    "sigmaZero": 1.0,
    "sigmaZeroExpl": "The surface tension (spring constant) of the
       fluid.",
    "snapshotFrequency": 1.0,
    "snapshotFrequencyExpl": "How often are snapshots saved. Bigger
       number can be used to get smoother animations, 0.0 means no
       snapshots.",
    "stdDevRad": 0.21,
    "stdDevRadExpl": "The standard deviation of the bubble radii.",
    "timeStepIn": 0.001,
    "timeStepInExpl": "The initial time step with which to increment
       the simulation time.",
    "wallDragStrength": 0.1,
    "wallDragStrengthExpl": "How strong the drag from the wall is.
       Should be between 0.0 and 1.0"
}
```

LISTING B.2: Inputs for phase separation simulations 1


```
{
    "avgRad": 1.0,
    "avgRadExpl": "The average radius of the bubbles.",
    "boxRelDim": {
        "x": 1.0,
        "y": 1.0,
        "z": 1.0
    },
    "boxRelDimExpl": "The relative dimension of the box.",
    "errorTolerance": 1e-05,
    "errorToleranceExpl": "The error tolerance level for the
        integration.",
    "flowLbb": {
        "x": 0.0,
        "y": 0.475,
        "z": 0.0
    },
    "flowLbbExpl": "(Relative) Lower-back-bottom of flow area.",
    "flowTfr": {
        "x": 1.0,
        "y": 0.525,
        "z": 0.0
    },
    "flowTfrExpl": "(Relative) Top-front-right of flow area.",
    "flowVel": {
        "x": 0.01,
        "y": 0.0,
        "z": 0.0
    },
    "flowVelExpl": "Flow velocity inside flow area.",
    "kParameter": 0.001,
    "kParameterExpl": "A bundle of constants related to gas exchange.
        Should be small relative to AvgRad.",
    "kappa": 0.1,
    "kappaExpl": "Relative strength of liquid-intermediated diffusion
        versus diffusion through thin films.",
    "maxDeltaEnergy": 1e-05,
```

```
    "maxDeltaEnergyExpl": "Maximum allowed energy change when relaxing
       the bubble bath.",
    "minNumBubbles": 2000,
    "minNumBubblesExpl": "The simulation runs as long as there are at
       least this many bubbles.",
    "muZero": 1.0,
    "muZeroExpl": "The viscosity of the fluid",
    "numBubblesIn": 1000000.0,
    "numBubblesInExpl": "Total number of bubbles to simulate.",
    "numBubblesPerCell": 32,
    "numBubblesPerCellExpl": "How many bubbles per cell.",
    "numStepsToRelax": 10000.0,
    "numStepsToRelaxExpl": "How many steps to take per 'relaxation
       round' when equilibrating the bubble bath.",
    "phiTarget": 0.851,
    "phiTargetExpl": "The target volume fraction.",
    "rngSeed": 13234,
    "rngSeedExpl": "The seed for the RNG.",
    "sigmaZero": 1.0,
    "sigmaZeroExpl": "The surface tension (spring constant) of the
       fluid.",
    "snapshotFrequency": 0.1,
    "snapshotFrequencyExpl": "How often are snapshots saved. Bigger
       number can be used to get smoother animations, 0.0 means no
       snapshots.",
    "stdDevRad": 0.21,
    "stdDevRadExpl": "The standard deviation of the bubble radii.",
    "timeStepIn": 0.001,
    "timeStepInExpl": "The initial time step with which to increment
       the simulation time.",
    "wallDragStrength": 0.1,
    "wallDragStrengthExpl": "How strong the drag from the wall is.
       Should be between 0.0 and 1.0"
}
```

Bibliography

[1] P. Stevenson. *Foam Engineering: Fundamentals and Applications*. Wiley, 2012.

[2] Shau-Tarng Lee and Natarajan S Ramesh. *Polymeric foams: mechanisms and materials*. CRC press, 2004.

[3] I.M. Lifshitz and V.V. Slyozov. The kinetics of precipitation from supersaturated solid solutions. *Journal of Physics and Chemistry of Solids*, 19:35–50, April 1961.

[4] Carl Wagner. Theorie der alterung von niederschlägen durch umlösen (ostwald-reifung). *Zeitschrift für Elektrochemie, Berichte der Bunsengesellschaft für physikalische Chemie*, 65(7-8):581–591, 1961.

[5] John von Neumann. Discussion – shape of metal grains. *Metal Interfaces*, 1952.

[6] B. S. Gardiner, B. Z. Dlugogorski, and G. J. Jameson. Coarsening of two- and three-dimensional wet polydisperse foams. *Philosophical Magazine A*, 80(4):981–1000, 2000.

[7] Kseniia Khakalo, Karsten Baumgarten, Brian P. Tighe, and Antti Puisto. Coarsening and mechanics in the bubble model for wet foams. *Phys. Rev. E*, 98:012607, Jul 2018.

[8] D. J. Durian. Foam mechanics at the bubble scale. *Phys. Rev. Lett.*, 75:4780–4783, Dec 1995.

[9] R. B. Potts. Some generalized order-disorder transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48(1), 1952.

[10] Gilberto L Thomas, RMC De Almeida, and François Graner. Coarsening of three-dimensional grains in crystals, or bubbles in dry foams, tends towards a universal, statistically scale-invariant regime. *Physical Review E*, 74(2):021407, 2006.

[11] Denis L Weaire and Stefan Hutzler. *The physics of foams*. Oxford University Press, 2001.

[12] Elizabeth A Holm, James A Glazier, David J Srolovitz, and Gary S Grest. Effects of lattice anisotropy and temperature on domain growth in the two-dimensional potts model. *Physical Review A*, 43(6):2662, 1991.

[13] Kenneth A Brakke. The surface evolver. *Experimental mathematics*, 1(2):141–165, 1992.

[14] SJ Cox. A viscous froth model for dry foams in the surface evolver. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 263(1-3):81–89, 2005.

[15] J A Glazier and D Weaire. The kinetics of cellular patterns. *Journal of Physics: Condensed Matter*, 4(8):1867, 1992.

[16] H. J. Frost and C. V. Thompson. Computer simulation of microstructural evolution in thin films. *Journal of Electronic Materials*, 17(5):447–458, Sep 1988.

[17] JP Kermode and D Weaire. 2d-froth: a program for the investigation of 2-dimensional froths. *Computer Physics Communications*, 60(1):75–109, 1990.

[18] D Weaire, JP Kermode, and J Wejchert. On the distribution of cell areas in a voronoi network. *Philosophical Magazine B*, 53(5):L101–L105, 1986.

[19] H Aref and T Herdtle. Topological fluid mechanics. In *Symposium Proceedings*, 1990.

[20] C. W. J. Beenakker. Evolution of two-dimensional soap-film networks. *Phys. Rev. Lett.*, 57:2454–2457, Nov 1986.

[21] Tohru Okuzono, Kyozi Kawasaki, and Tatsuzo Nagai. Rheology of random foams. *Journal of Rheology*, 37(4):571–586, 1993.

[22] Tohru Okuzono and Kyozi Kawasaki. Intermittent flow behavior of random foams: a computer experiment on foam rheology. *Physical Review E*, 51(2):1246, 1995.

[23] Robert Lemlich. Prediction of changes in bubble size distribution due to interbubble gas diffusion in foam. *Industrial & Engineering Chemistry Fundamentals*, 17(2):89–93, 1978.

[24] Victor Eijkhout, Robert van de Geijn, and Edmond Chow. *Introduction to High Performance Scientific Computing*. Zenodo, April 2016. Book source and printed copies are available: http ://pages.tacc.utexas.edu/ eijkhout/istc/istc.html.

[25] NVIDIA Corporation. CUDA C Programming Guide.

[26] Gijs Katgert and Martin van Hecke. Jamming and geometry of two-dimensional foams. *EPL (Europhysics Letters)*, 92(3):34002, 2010.

[27] Andrea J Liu and Sidney R Nagel. Nonlinear dynamics: Jamming is not just cool any more. *Nature*, 396(6706):21, 1998.

[28] Hernán A Makse, David L Johnson, and Lawrence M Schwartz. Packing of compressible granular materials. *Physical review letters*, 84(18):4160, 2000.

[29] Farhang Radjai, DE Wolf, Stéphane Roux, Michel Jean, and Jean Jacques Moreau. Force networks in dense granular media. 1997.

[30] Pierre Dantu. Étude expérimentale d'un milieu pulvérulent compris entre deux plans verticaux et parallèles. *Annales des Ponts et Chaussées, IV*, pages 193–202, 1967.

[31] Antoinette Tordesillas and Maya Muthuswamy. On the modeling of confined buckling of force chains. *Journal of the Mechanics and Physics of Solids*, 57(4):706–727, 2009.